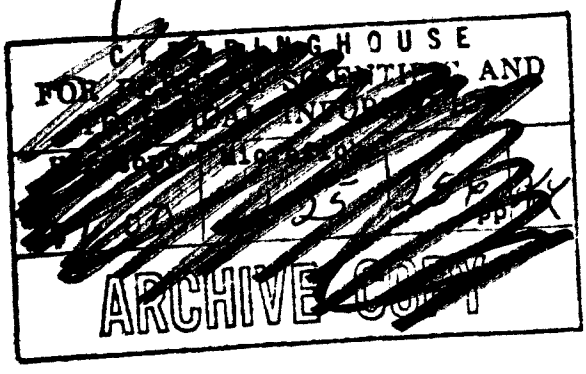
D.I.A.



IDHS 1410 FORMATTED FILE SYSTEM

(IDHS 1410 FFS)

FILE MAINTENANCE AND FILE GENERATION MANUAL

REVISED 1 AUGUST 1966

TABLE OF CONTENTS

Page

SECTION ONE - INTRODUCTION

1-1. General ---------- 1-1
1-2. The 1410 Formatted File System ---------- 1-2
1-3. The 1410/7010 Operating System ---------- 1-3
1-4. The Formatted File System Programs ---------- 1-3
1-5. Introduction to this Document ---------- 1-4
Purpose ---------- 1-4
Structure ---------- 1-5

SECTION TWO - FILE CONCEPTS

2-1. General ---------- 2-1
The Formatted File ---------- 2-1
The Data ---------- 2-1
2-2. The File Record ---------- 2-2
Fields ---------- 2-2
Groups ---------- 2-2
Periodic Subsets ---------- 2-3
Periodic Sets ---------- 2-3
Fixed Set ---------- 2-3
Variable Set ---------- 2-3
2-3. File Record Format ---------- 2-4
Record Control Group (Record ID) ---------- 2-4
2-4. The CMFLA Sample File ---------- 2-6
General ---------- 2-6
The Fixed Set ---------- 2-6
Periodic Set One ---------- 2-6
Periodic Set Two ---------- 2-6
Variable Set ---------- 2-6
Description of Elements ---------- 2-8
Synonyms ---------- 2-8
2-5. Purpose of File Generation and File Maintenance ---------- 2-8
2-6. Special Geographic Operator ---------- 2-12
2-7. File Generation ---------- 2-12
File Structuring ---------- 2-12
File Revision ---------- 2-13
2-8. File Maintenance ---------- 2-13
General ---------- 2-13
File Maintenance Supervisor Phase ---------- 2-15
Input Processing Phase ---------- 2-16
FM Sort Phase ---------- 2-16
File Maintenance Proper Phase ---------- 2-17

Page

FMX - Purge Routine for Subset Deletion ---------- 2-18
Detailed Description ---------- 2-18
Cross-Index Updater Phase ---------- 2-19
Need for Cross-Indexing a File ---------- 2-19
Eliminating Record ID's in the Cross Index From Consideration Prior to Accessing the File ---------- 2-21
How to Index a File and Specify the Elimination of Fields ---------- 2-21
Purpose of the Cross-Index Updater Phase ---------- 2-21
-9. Section Two Summary ---------- 2-22

ECTION THREE - ILE GENERATION

-1. File Structuring ---------- 3-1
General ---------- 3-1
Quantitative Limits ---------- 3-4
Data Element Placement ---------- 3-5
Variable Set Data ---------- 3-5
FS Input Card Specifications ---------- 3-6
Common Elements ---------- 3-6
Detailed Input Card Formats ---------- 3-6
Editing ---------- 3-10
Insertion ---------- 3-12
Zero Suppression ---------- 3-13
Asterisk Protection ---------- 3-14
Floating Dollar Sign ---------- 3-15
Sign Control ---------- 3-16
Decimal Control ---------- 3-17
Edit Card ---------- 3-18
Subroutine Verification ---------- 3-19
SUB and TAB Cards ---------- 3-20
FIELD Card ---------- 3-21
GROUP Card ---------- 3-24
VSET Card ---------- 3-25
INDEX Card ---------- 3-26
ELIM Card ---------- 3-28
GEOOP Card ---------- 3-30
Allowable Card Sequence ---------- 3-31
The CMF File Structuring Run Example ---------- 3-34
The File Format Table ---------- 3-34
Logical Record 1 ---------- 3-34
Logical Record 2 ---------- 3-34
Logical Record 3 ---------- 3-37
Logical Record 4 ---------- 3-37
Logical Record 5 ---------- 3-38
Logical Record 6 ---------- 3-38
Logical Record 7 ---------- 3-38
Logical Record 8 ---------- 3-39
Logical Record 9 ---------- 3-40

Page

File Structuring Error Comments With Explanations ------ 3-51
3-2. File Structuring Summary ------ 3-61
3-3. File Revision ------ 3-63
General ------ 3-63
FR Input Card Specifications ------ 3-67
FR Control Cards ------ 3-67
FR Change Cards ------ 3-68
A Change of Field Size (S) ------ 3-69
A Change of Field Name (N) ------ 3-69
A Change of Field Name and Size (B) ------ 3-70
Addition of a Periodic Set or a Variable Set (Z)- 3-70
Change of Periodic Set ID Number (C) ------ 3-71
Use of "Interim" FFT's ------ 3-72
File Revision Examples, Using the CMFLA File ------ 3-73
Example #1 ------ 3-74
Example #2 ------ 3-84
File Revision Error Comments, With Explanations ------ 3-92
Explanation of Abbreviations Used in Actual Changes Listing ------ 3-95
3-4. File Revision Summary ------ 3-96

SECTION FOUR - FILE MAINTENANCE

4-1. General ------ 4-1
4-2. General Operation of Each Phase of File Maintenance ------ 4-1
Change Record ID - Then Resort Data File ------ 4-8
Entering Data ------ 4-10
Transaction Which May Be Performed ------ 4-10
Input Format Types ------ 4-11
Internal Format ------ 4-11
External Format ------ 4-11
4-3. Specifications for Internal Format Input ------ 4-12
Specifications for the Input Control Card (For Internal Format) ------ 4-12
Specifications for the Internal Format Input Data Card - 4-13
4-4. Additional Comments Concerning the Data Entry in the I.F. Input Data Card ------ 4-15
Numeric Fields ------ 4-15
Alphameric Fields ------ 4-15
Maximum Data Length ------ 4-15
4-5. Specifications for the Ippakend Card for Internal Format ------ 4-16
4-6. Operation Codes ------ 4-16
4-7. Design of an Input File for External Format ------ 4-19
General ------ 4-19
Creation of an External Format Input File ------ 4-19
Using Existing Files of Data as E.F. Input Files for FFS Data File Creation ------ 4-20
4-8. General Input File Requirements and Definition of Terms ------ 4-20
Input File ------ 4-20

Input Record ---------- 4-21
Input Record Type ---------- 4-21
Input Record Type Code ---------- 4-21
Input Group ---------- 4-21
Input Group Control Field ---------- 4-21
Record ID (Record Control Group) ---------- 4-22
4-9. Preferred Location of Control Fields on Input Records ---------- 4-25
4-10. Specific Input File Requirements (External Format) ---------- 4-25
Fixed Fields ---------- 4-25
Splitting Fixed Fields ---------- 4-25
Fixed Groups ---------- 4-25
Splitting Fixed Groups ---------- 4-26
Periodic Fields/Groups ---------- 4-26
Periodic Subset Sequence Number (PSSQn) ---------- 4-27
Periodic Fields/Groups - Split ---------- 4-30
Subset Entries - Multiple and Single ---------- 4-30
Variable Set ---------- 4-33
4-11. Using External Format Input to Update FFS Data Files ---------- 4-35
ICC Opcode Method ---------- 4-35
(Input) Record Opcode Method ---------- 4-35
Record Opcode Specifications ---------- 4-36
Limitations of Both Opcode Methods ---------- 4-36
Record Opcode Method Limitations ---------- 4-36
ICC Opcode Method Limitation ---------- 4-37
Multiple Usage of Input Record Elements ---------- 4-37
4-12. Specifications for External Format Input ---------- 4-37
Specifications for the Input Control Card (For External Format) ---------- 4-37
Specification for the Input Descriptor Deck ---------- 4-41
Control Location Section ---------- 4-41
Field Extraction Section ---------- 4-41
Input Record Type Code Locator Card Specifications ----- 4-41
Input Group Control Field Locator Card Specifications -- 4-44
Opcode Position Locator Card Specifications ---------- 4-46
Record Control Field Locator Card Specifications ------- 4-49
Additional Comments Concerning the Record Control Field Locator Card ---------- 4-51
Field Extract Card Specifications ---------- 4-52
(IDD) End Card Specifications ---------- 4-56
Additional Comments About the IDD ---------- 4-56
4-13. The Transaction Record Layout ---------- 4-58
General ---------- 4-58
Transaction Confirmation Listing ---------- 4-62
4-14. Input Processor Phase Error Messages ---------- 4-62
IP Error Specifications ---------- 4-62
Delete Entire Run (System Errors) ---------- 4-63
Delete Job ---------- 4-63
Delete Field From Input Format Table ---------- 4-63

Page

Do Not Process Input Record But Print It ---------- 4-64
Do Not Process This Data But Print It Out (With Transaction Record) ---------- 4-64
Input Processor Error Comments Listing, With Explanations ---------- 4-64
4-15: Transaction Confirmation Listing - Error Flags ---------- 4-69
BLAN ---------- 4-69
CREA ---------- 4-69
DBLA ---------- 4-69
ERRN ---------- 4-69
MAXL ---------- 4-69
NADD ---------- 4-70
NCAD ---------- 4-70
NCRF ---------- 4-70
NOCR ---------- 4-70
NOID ---------- 4-70
NOOT ---------- 4-70
NSET ---------- 4-70
NTYP ---------- 4-71
NVST ---------- 4-71
PSER ---------- 4-71
RCTL ---------- 4-71
SORT ---------- 4-71
SSID ---------- 4-71
4-16. Cross-Index Updater Phase Error Comments Listing, With Explanations ---------- 4-71
4-17. External Format Input Examples Using the CMFLA Sample File ---- 4-72

LOGICAL FILE MAINTENANCE

4-18. Why Logical Maintenance ---------- 4-76
4-19. Subsystem Operation ---------- 4-76
4-20. Control Card Entries ---------- 4-76
4-21. Types of LFM Runs ---------- 4-81
4-22. Modes of Operation ---------- 4-81
Example One - The File in Error ---------- 4-83
Example Two - Parameter Changes ---------- 4-83
Example Three - "Automatic" Update ---------- 4-84
Example Four - Reducing Input ---------- 4-85

APPENDIXES

Appendix A - Glossary of Terms A-1

Appendix B - Abbreviations Used B-1

Appendix C - General Description of the 1410/7010 Operating System C-1

Appendix D - Character Definition Table D-1

Appendix E - Math Formula to Determine the Maximum Number of Fields/ Groups Allowed for Your File Under FFS E-1

ILLUSTRATIONS

FIGURE	TITLE	PAGE
1-1	Major Components of the 1410 FFS, Mark II	1-2
2-1	1410 FFS Fil Record Layout	2-5
2-2	Format of th CMFLA Sample File	2-7
2-3	Fixed Field Definition - CMFLA File	2-9
2-4	Periodic Set One, Field Definition - CMFLA File	2-10
2-5	Periodic Set Two, Field Definition - CMFLA File	2-11
2-6	The 1410 Formatted File System (Mark II)	2-14
3-1	General Operation of the File Structuring Phase	3-3
3-2	FSJOB and ENDFS Cards	3-7
3-3	BYPASS and * (comments) cards	3-9
3-4	EDIT Card	3-18
3-5	SUB and TAB Cards	3-20
3-6	FIELD Card	3-21
	FIELD Card (continued)	3-22
	FIELD Card (continued)	3-23
3-7	GROUP Card	3-24
3-8	VSET Card	3-25
3-9	INDEX Card	3-26
	INDEX Card (continued)	3-27
3-10	ELIM Card	3-29
3-11	GEOOP Card	3-30
3-11A	"Cards Which May Immediately Follow" Table	3-32
3-11B	"Cards Which May Precede" Table	3-33
3-12	FFT Logical Record Layout	3-35
3-13	FS Run Example	3-41
3-14	FS Run Example	3-42
3-15	FS Run Example	3-43
3-16	FS Run Example	3-44
3-17	FS Run Example	3-45
3-18	FS Run Example	3-46
3-19	FS Run Example	3-47
3-20	FS Run Example	3-48
3-21	FS Run Example	3-49
3-22	FS Run Example	3-50
3-23	General Operation of the File Revision Phase	3-66
3-24	Original Configuration of "Take-off Date/Time" Data	3-72
3-25	Desired Configuration of "Take-off Date/Time" Data	3-72
3-26	Relationship of Old, Interim, and New FFTs	3-73
3-27	Cards for Example #1	3-75
3-28	Preliminary Version of the CMFLA File Before Revision By Example #1	3-76
3-29	FS Input Cards for "Old FFT" for FR Example #1	3-77
3-30	FS Input Cards for "Old FFT" for FR Example #1	3-78
3-31	FS Input Cards for "Old FFT" for FR Example #1	3-79

FIGURE	TITLE	PAGE
3-32	Preliminary Version of the CMFLA File After Revision by Example #1, and Prior to Revision by Example #2	3-80
3-33	FS Input Cards for "New FFT" of FR Example #1, and "Old FFT" for FR Example #2	3-81
3-34	FS Input Cards for "New FFT" of FR Example #1, and "Old FFT" for FR Example #2	3-82
3-35	FS Input Cards for "New FFT" of FR Example #1, and "Old FFT" for FR Example #2	3-83
3-36	Cards for Example #2	3-85
3-38	The CMFLA File After Revision By Both Examples #1 and #2	3-86
3-39	FS Input Cards for "New FFT" for FR Example #2	3-87
3-40	FS Input Cards for "New FFT" for Example #2	3-88
3-41	FS Input Cards for "New FFT" for Example #2	3-89
3-42	FS Input Cards for "New FFT" for Example #2	3-90
3-43	FS Input Card. for "New FFT" for Example #2	3-91
3-44	Abbreviations Used in Actual Changes Listing	3-95
4-0	General Operation of FM Supervisor	4-2
4-1	General Operation of Input Processing	4-3
4-2	General Operation of FM Sort	4-4
4-3	General Operation of FM Proper	4-5
4-4	General Operation of Cross-Index Updater	4-6
4-5	General Operation of FMX	4-7
4-6	ICC For Internal Format	4-12
4-7	Internal Format Input Data Card	4-14
4-8	Summary of Legal Operations That May Be Specified For Elements of a File Record	4-17
4-9	The Input Control Card for External Format	4-39
4-9	The Input Control Card for External Format (continued)	4-40
4-10	The Input Record Type Code Locator Card	4-43
4-11	The Input Group Control Field Locator Card	4-45
4-12	The Op Code Position Locator Card	4-47
4-12	The Op Code Position Locator Card (continued)	4-48
4-13	The Record Control Field Locator Card	4-50
4-14	The Field Extract Card	4-54
4-14	The Field Extract Card (continued)	4-55
4-15	IDD Card Formats	4-57
4-16	Transaction Record Layout	4-59
4-17	External Format Input Sample, Example #1	4-73
4-18	External Format Input Sample, Example #2	4-74
4-19	External Format Input Sample, Example #3	4-75

SECTION ONE

INTRODUCTION

1-1. GENERAL:

a. Industry today depends upon the accurate analysis of large amounts of multisource information which must be brought together, collated, evaluated, categorized, and filed in such a manner that the rapid retrieval of specific items by a variety of systems is possible. All of this must be done quickly and thoroughly if the information is to be of maximum use for decision making. The difficulty and time required to assemble, collate, digest and evaluate the information collected increases considerably when there is only a modest increase in the volume, types, and sources of information. As the demands placed upon the analyst multiply (in terms of quality, quantity, and comprehensiveness), the time he can allot to researching a problem and assembling material for analysis becomes critical. In conjunction with the above factors, the inherent slowness of the manual approach to the analysis of multisource information results in a need for automation. A list of some of the operations which lend themselves to high-speed automation includes editing, sorting, merging, matching, collating, indexing, filing, updating, cross-referencing, purging, retrieving, summarizing, report generating, and all types of mathematical computation and logical processes. The automation of these and similar operations not only allows the analyst more time for evaluation, but also enables a much more detailed and comprehensive analysis of quantities of information that could not be undertaken by manual techniques.

b. The flexibility, vast high-density storage capacity, and processing ability coupled with today's high-speed electronic data processing systems enables users to satisfy the requirements of automating much of the development of their reports. The 1410 Data Processing System (DPS) coupled with the 1410 Formatted File System (FFS) increases the comprehensiveness and accuracy and greatly accelerates the development of reports from the data.

1-2. THE 1410 FORMATTED FILE SYSTEM: The 1410 Formatted File System is a general purpose, flexible data handling system. It consists of a collection of specially developed programs coupled with the 1410/7010 Operating System. Figure 1 - 1. illustrates the major components of the 1410 FFS.

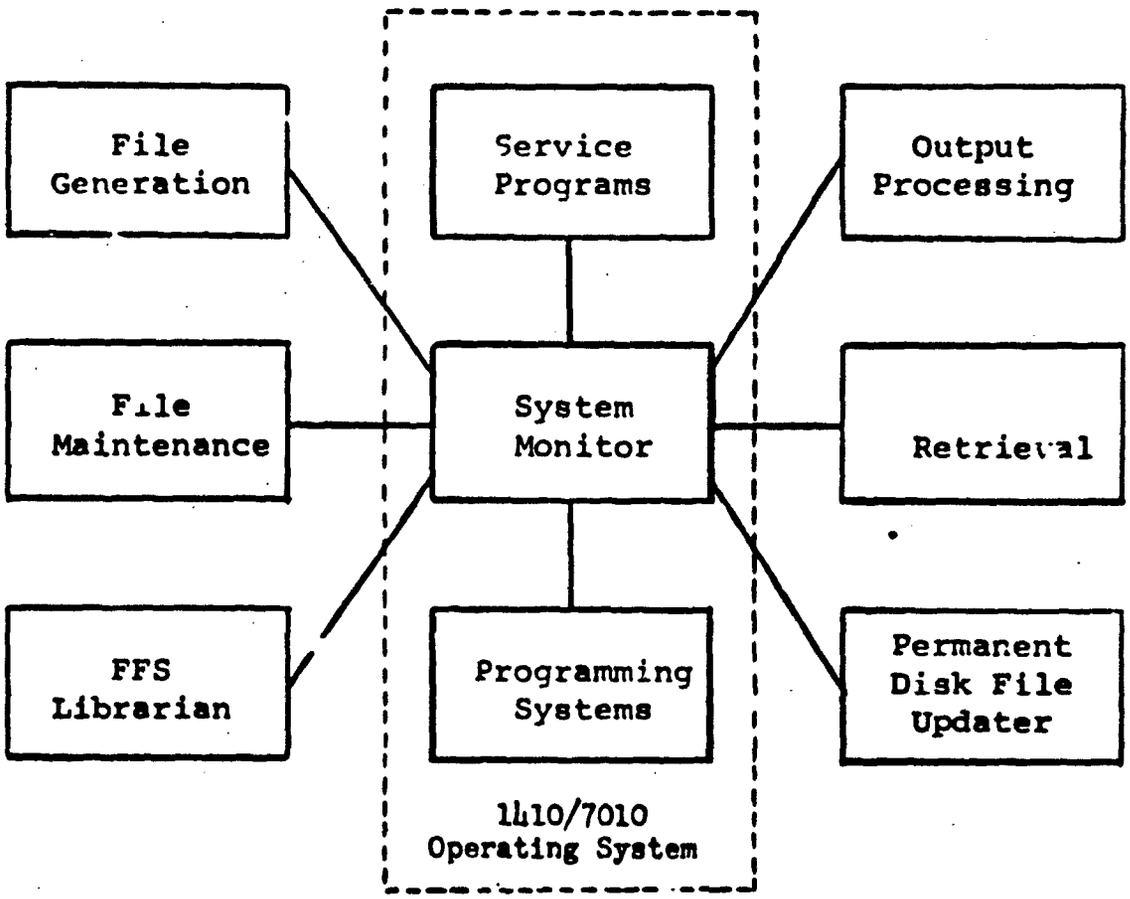


Figure 1 - 1. Major Components of The 1410 Formatted File System (FFS),

1-3. THE 1410/7010 OPERATING SYSTEM:

a. The 1410/7010 Operating System is an integrated set of programs and programming systems that finds widespread and diverse applications whenever the 1410 or 7010 Data Processing Systems are used. The fundamental purpose of the operating system is to enable the writing, assembling, and execution of programs with a minimum expenditure of programmer time, machine-operator time, and machine time. These savings in time are accomplished by means of:

Programming Systems (COBOL, FORTRAN, Autocoder)

Service Programs (Input/Output Control System, Tape Sort, System Generation, Teleprocessing Supervisor, Random Processing Scheduler, Utility Programs)

System Monitor (Resident Monitor, Linkage Loader, Transitional Monitor)

b. One of the purposes of system monitor is to control the sequencing and monitoring of the programs which are an integral part of the operating system itself, as well as the Formatted File System programs which operate within the framework of the operating system. As the heart of the operating system, the system monitor performs other major functions, such as the assignment of input/output units, transition between jobs, program loading and relocation, and the linkage of independently compiled programs. Communication with the operator is also accomplished by system monitor for such things as errors or required setups for programs associated with the operating system. Operating procedures are standardized and simplified as much as practical.

c. Appendix C of this manual contains a general description of the 1410/7010 Operating System for the reader desiring more information.

1-4. THE FORMATTED FILE SYSTEM PROGRAMS:

a. Each of the special purpose Formatted File System programs is designed to operate within the framework of the 1410/7010 Operating System. Although each of the Formatted File System programs is itself a consolidation of several smaller programs, it is convenient to group them into the six following functional areas:

File Generation

File Maintenance

FFS Librarian

Permanent Disk File Updater

Retrieval

Output Processing

b. The function of each of the Formatted File System programs is very briefly described below:

(1) File Generation. Structures new files, restructures existing files, and provides for file data rearrangement from the previous file structure to the new file structure.

(2) File Maintenance. Maintains the data content of the files by changing, adding to, or deleting existing data entries; or creating new data entries in the files. The input data supplied by the user may be extracted, converted, edited, etc., as specified by the user. File maintenance may also provide a "table of contents" (cross index and file data table) for each file it maintains. A record of all maintenance performed is generated for confirmation purposes.

(3) FFS Librarian. Keeps the items which are stored in the FFS relocatable execution library up to date. Some of these items are file format tables, report instruction tables, and input and output conversion subroutines.

(4) Permanent Disk File Updater. Keeps the items which are stored on permanent disk up to date; these items include file data tables and cross indexes.

(5) Retrieval. Selectively obtains information from the files, based upon parameters specified in reports and/or requests supplied by the user. The information retrieved may be sorted by selected data content, as well as by various other criteria.

(6) Output Processing. Structures report formats based upon specifications supplied by the user. The information retrieved from the files by retrieval is extracted, edited, converted and formatted as specified, and output in the form of reports. A variety of counts, totals, and computations may also be performed and included in the report. Also provided are capabilities for outputting directly from the data file, as well as outputting the transaction confirmation tape from file maintenance.

1-5. INTRODUCTION TO THIS DOCUMENT:

a. Purpose. This manual is specifically directed toward users who perform one or more of the following data base support functions in connection with the IDHS 1410 FFS.

Helps design FFS data files.

Structures FFS data files.

Makes structural changes to existing FFS data files.

Helps design formats for input data to FFS data files.

Generates the descriptive and control specifications required by the system for input data to FFS data files.

Maintains the data content of FFS data files.

The 1410 FFS is described to the user in sufficient detail to enable him to accomplish the above tasks within the framework of the system. Although the description includes adequate theory, explanations, specifications, and examples for the needs of the above mentioned users, this document is not program oriented and does not provide an analysis of the logic of the program segments comprising the 1410 FFS. Such information is contained in the "Program System Description" Manual for the 1410 FFS.

b. Structure. This document is presented in four primary sections and five appendixes:

Section I, INTRODUCTION, gives the need for and application of automation to help develop the user's report, brief description of the 1410 FFS and its major components, purpose of the document, users for whom it is intended, and its structure.

Section II, FILF CONCEPTS, describes the concept of a formatted file, the characteristics of fixed and periodic data, and the makeup of fixed and periodic fields. The various methods of categorizing data are defined and related (i.e., fixed and periodic fields, groups, sets, periodic subsets, variable set, etc.) in terms of a file record. The sample file CMFLA is introduced and defined to illustrate the above. Each phase of the file generation and file maintenance programs of the 1410 FFS is described in general terms.

Section III, FILE GENERATION, describes the procedures, control cards, data formats, and provides other information necessary to structure new and revise existing files. The inter-relationship between file structuring and file revision is shown and they are differentiated from file maintenance. Detailed card formats are presented along with instructions for their use. The printouts listing inputs and outputs from a file structuring run to create the sample file CMFLA are shown. A complete listing of all error messages that can be logged on the printer during file revision or file structuring is included, with explanations where required.

Section IV, FILE MAINTENANCE, describes the procedures, control cards, and data input requirements, and provides other information necessary to maintain the data content of FFS data files. Both internal and external input formats are described in detail. Detailed card formats are presented along with instructions for their use. The CMFLA file is again used as an illustrative example. A complete listing of all error messages that can be logged on the printer during any of the phases of file maintenance is given, by phase, with explanations where necessary.

Appendix A, GLOSSARY OF TERMS

Appendix B, ABBREVIATIONS USED

Appendix C, GENERAL DESCRIPTION OF THE 1410/7010 OPERATING SYSTEM

Appendix D, CHARACTER DEFINITION TABLE

Appendix E, Math Formula to determine the maximum number of field/ :oups allowed for your file under FFS.

SECTION TWO

FILE CONCEPTS

2-1. GENERAL:

a. The Formatted File.

(1) Large amounts of multisource information covering a broad range of subjects and received at the data processing complex in varied forms. This information is categorized by subject and/or application and added to a collection of similarly categorized data called a file (or data file).

(2) The file has a definite structure or pattern called the file format which facilitates the placing of elements of information in the file and their subsequent retrieval. When the elements of information are arranged according to the file format, each arrangement contains a description of an activity, event, objective, person, place, thing, etc. Each such arrangement of elements of information is called a file record (or data record).

(3) Thus a formatted file is an ordered collection of related file records, each of which contains data arranged according to a previously established file format.

b. The Data.

(1) The values applied to some elements of the description of an event or activity do not change over the span of space or time covered by a file record. However, there may be other elements which assume several values within the span of the event covered by the file record. The file record must include each of the values attained by such varying elements in order to contain a complete and accurate history or description. The elements whose values do not change during the span ordinarily need be recorded only once in the file record. Provision must be made for recording the changing elements several times.

(2) The data which describes the unchanging elements during the span of a file record is called fixed d[illegible] The file format allows for the insertion of one value for each of these [illegible]xed data elements in a file record. The data which may attain several values for the same descriptive element is called periodic data. The file format allows for a number of entries in each periodic data element in the file record.

(3) In addition to fixed and periodic data there is a type of data which may be categorized as remarks or comments related to the event described by the file record. This type of data is quite variable in content and size from one file record to another, and might be appended to those file records requiring such remarks or comments.

2-2. THE FILE RECORD:

a. Fields. The file record is a collection of elements of data arranged in the pattern specified by the file format. The smallest unit of data is the field. Each field has a defined length and contains only one specific type of data. If the data content of the field is fixed data, the field is a fixed field and will appear only once within the file record. If the data content of the field is periodic data, the field is a periodic field and may appear many times within the file record. Assume that the following report were received:

> American Airlines DC8B, passenger flight #123 from Boston to San Diego completed with stops in Cleveland, Omaha, Salt Lake. LV Boston 2312, 25 DEC 64. AR Cleveland 0037. Altitude 35500. LV Cleveland 0054, 26 DEC 64. AR Omaha 0220. Altitude 33500. LV Omaha 0236, 26 DEC 64. AR Salt Lake 0425. Altitude 35500. LV Salt Lake 0440, 26 DEC 64. AR San Diego 0610. Altitude 31500.

(1) The report concerns a commercial flight (an event) from Boston to San Diego and contains information which can be broken up into several fields such as airline name, flight number, point of departure, and point of destination. Each type of data in this report will be used in a specific field of the file record.

(2) Types of information such as flight number, aircraft type, and airline name are unchanging during the event and are classified as information belonging in the fixed field category.

(3) Some of the information types are repeated within the report. For example, there are four locations from which a leg of the flight originated: Boston, Cleveland, Omaha, and Salt Lake City. This information changes during the event and is placed in periodic fields. The periodic field named LEG ORIGIN would appear four times in the file record, once for each of the places from which a takeoff occurred. If more stops and more legs were given in the report, the number of periodic fields would increase. This characteristic of periodic fields to appear more than once within a file record allows the length of a file record to be variable even though field lengths are fixed.

b. Groups. Within the file record there can exist a closer relationship between some of the fields than exists between other fields. For example, in the report described above data on the date and time of each takeoff is included. To accommodate this data, the file record could contain two periodic fields named DATE and TIME. However, because of the nature of the data in these two fields, they are often reported, retrieved, and manipulated as a unit, called a group. A group is a collection of two or more adjacent fields within a file record which may be treated as a single data entity. The fields within a group do not lose their individual identities because of this grouping and may be treated like any other field. Groups are named just as are fields, and a title such as TAKEOFF DATE/TIME

might be appropriate in this case. Groups are categorized as either periodic groups or fixed groups, depending upon which type of fields are grouped. Both fixed and periodic fields cannot be included within the same group.

c. Periodic Subsets. The change of a value recorded in one periodic field is usually accompanied by changes in other associated periodic fields. In our example, the number of passengers, altitude, and gross weight are likely to change in each leg of the flight. Periodic fields containing information which is related in this manner are consolidated into periodic units called periodic subsets. When periodic data is added to a file record it is usually added in units of periodic subsets rather than individual periodic fields or groups. (This is not meant to imply that individual fields or groups in an existing subset cannot be changed for correction or updating.) The periodic subset structure is specified by the file format, while provision is made for as many identically structured periodic subsets as is required.

d. Periodic Sets.

(1) In some circumstances, certain periodic data changes which occur in the history of an event are not directly associated with other periodic data changes. For example, at some of the stops in the flight, unscheduled maintenance functions may be performed. The maintenance activities are not releted to, or normally associated with the altitude, number of passengers, or the gross weight, etc. during each leg of the flight. The periodic data concerning maintenance activities may be entered into separate periodic subsets which have a different structure than the periodic subsets containing data about each leg of the flight.

(2) Periodic subsets having identical formats are grouped together into periodic sets. Thus each periodic subset within a periodic set contains the same fields and groups in the same order, with only the data content of the periodic subsets varying. Periodic subsets having different formats are grouped into different periodic sets. The number of periodic subsets that may be in one periodic set is limited to 599, or the lesser number which causes the limitation on the size of the file record itself to be exceeded. The number of periodic sets per file record is limited to a maximum of eight.

e. Fixed Set. The collection of all the fixed fields of a file record is often called the fixed set. Since fixed fields are nonrepetitive within a file record there is always one and only one fixed set in any file record. The first data fields of a file record are always the fixed fields, which comprise the fixed set. After the fixed set comes the periodic set(s), if any. Next to appear is the variable set which is described next.

f. Variable Set. The 1410 FFS allows for one additional type of data to be contained in a file record. It is data which cannot be readily formatted and is typified by remarks or comments which are often

added locally and not received from the ordinary sources. This unformatted data is entered into a set called the variable set. The variable set, unlike the other sets, can contain only one field of variable length. This is the only field in the file record which is not of fixed length. There can only be one variable set/field per file record. If a file record has a variable set, it appears after the periodic set(s), if any, in the file record. The data content of the variable set cannot be utilized as a parameter or factor for retrieval or output processing, as may the data content of the fixed and periodic sets. However, the data content of the variable set may be retrieved and output, as may the data content of the fixed and periodic sets.

2-3. FILE RECORD FORMAT:

a. Figure 2-1 illustrates the layout of a file record. The elements outlined with dashed lines contain control information only and are program maintained.

b. One point illustrated by figure 2-1 is the inclusion of a field in more than one group. Notice periodic field 2E in each of the periodic subsets of periodic set 2. It is included in periodic group 2F as well as 2H. Thus field 2E may be referenced by referring to:

Field 2E

Group 2F (Field 2E accompanied by 2D)

Group 2H (Field 2E accompanied by 2G)

There is no practical limitation on the number of groups within which a field may be defined.

c. Record Control Group (Record ID)

(1) Some means must be provided for distinguishing one file record from another, so that each may have a unique identity. In the 1410 FFS this is accomplished by each file record having unique data content in a special group called the record control group or "record ID." This record control group consists of one or more initial fields of the fixed set. The file designer insures that the characteristics of the sum of the information in the record control group is such that no two file records will ever have identical data content in their record control groups. In figure 2-1, fixed fields FA and FB are the components of the fixed group FC which is the record control group or record ID.

(2) The order in which the file records are kept in the data file is also controlled by the record control group. The file records are ordered in ascending sequence by the data content of their record control groups. The record control group may be thought of as a file's sort key; major to minor -- left to right. The maximum number of characters that may be included in the record control group is 30.

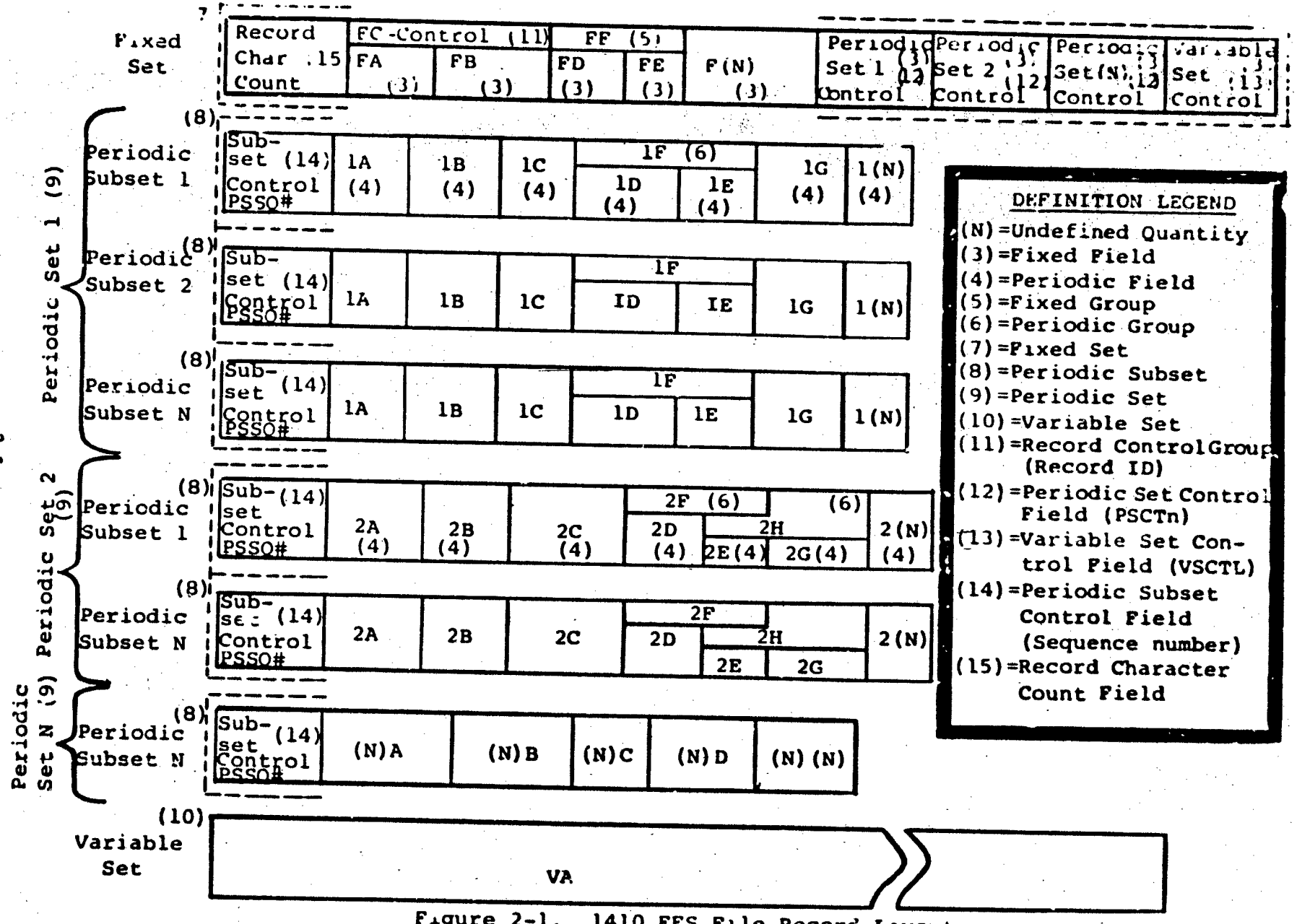


Figure 2-1. 1410 FFS File Record Layout

2-4. THE CMFLA SAMPLE FILE:

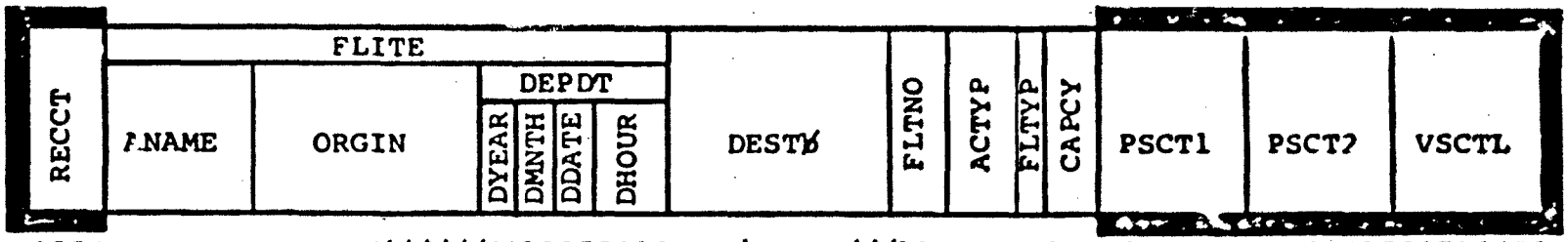
a. General. To allow better illustration of the concepts of file generation and maintenance, and to provide continuity between subjects, a sample file is used throughout this document. This hypothetical file is named the Commercial Flight File. The file mnemonic is CMFLA. (File mnemonics are five characters long, the last character being an A.) The purpose of this file is to maintain a record of commercial airline flights made within the United States. The structure of the sample file is such that one file record exists for each flight. A flight is considered to be from takeoff at the origin point until disembarkation at the final destination point of the flight. The structure of file records within the file is shown in figure 2-2. The nondata fields have a dark border around them.

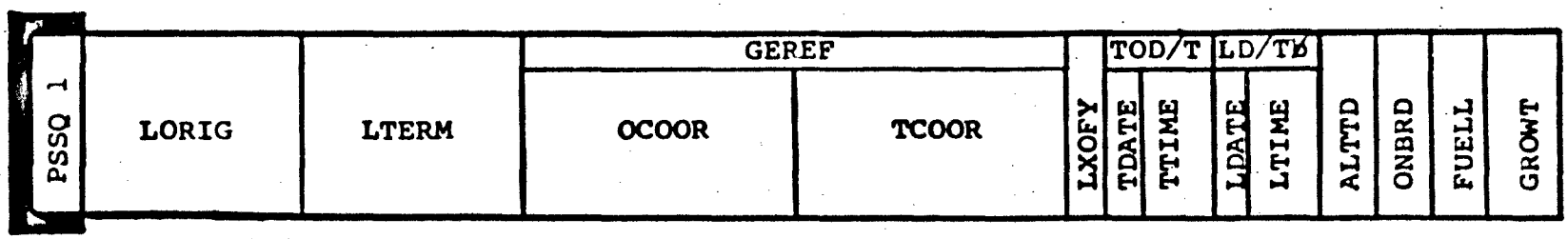
b. The Fixed Set. The data given in the fixed set occurs once per flight and is not subject to change throughout the flight. Notice that the record control group (record ID) called flight contains six fields, the sum of which is thirty characters. The last four of these fields are also in a fixed group called scheduled departure which is entirely contained within the record control group. If the assumption is valid that no airline will have more than one flight departure from the same airport at exactly the same time, then no two file records in the CMFLA file will have the same data content in their record control groups. Thus, each may be uniquely identified by that data content.

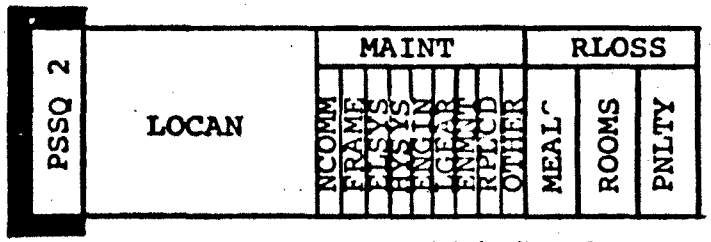
c. Periodic Set One. Periodic set one of the CMFLA file gives information on each leg of the flight. (Leg is used here to indicate each portion of the flight began by a takeoff and terminated by a landing.) This data is repeated once per leg, and all of this data is likely to change from leg to leg. In the case of a nonstop flight from Denver to Los Angeles there would be only one periodic subset in periodic set one. A flight from Boston to San Diego, with stops in Cleveland, Omaha, and Salt Lake City would have four periodic subsets in periodic set one.

d. Periodic Set Two. Periodic set two has a subset entry for each location at which unscheduled maintenance occurs, during the course of the flight. A given file record may have one or more subsets in periodic set two, or it may have none at all.

e. Variable Set. The variable set is used for any remarks or comments desired. One of an almost limitless number of possibilities follows: Assume that at the completion of the first leg of a flight, serious trouble occurred in the electrical system causing the flight to be continued on a different type of aircraft. The variable set might be used to indicate the new aircraft type, its capacity, as well as any other remarks desired.







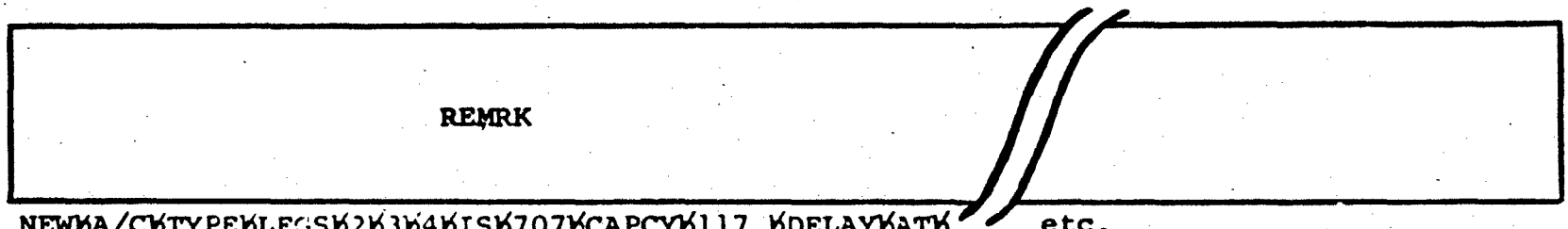


Figure 2-2. Format of the CMFLA Sample File.

f. Description of Elements. Figures 2-3, 2-4, and 2-5 provide a detailed description and explanation of each field and group of a CMFLA file record. Figure 2-3 describes the fields in the fixed set, 2-4 the fields in periodic set one, and 2-5 the fields in periodic set two.

g. Synonyms. In order to save space, field and group mnemonics are limited to five characters in length. When the file is designed, five-character names which approximate the user's label are assigned to each element. To avoid the necessity of having the analysts and retrieval technicians learn all of the element mnemonics, (which are sometimes rather cryptic), retrieval allows the use of synonyms. These synonyms are meaningful words which are more familiar and more easily remembered than the element mnemonics. The synonyms are constructed and defined externally to the system, and are then placed on the FFS library as a synonym table by the FFS Librarian. An element may have more than one associated synonym, which allows a more flexible retrieval language, since any one of several names may be used to specify an individual parameter. When designing a file, an effort should be made to insure that an element's output label*, mnemonic, and synonym(s) are as much alike as practical. In the ideal case the mnemonic is not ambiguous and the output label is identical to the mnemonic. In this case no synonym is required or desirable. When the five-character limitation on element mnemonics prevents such an ideal case, it is usually possible and desirable to have an identical output label and synonym. It can be somewhat confusing when an element has multiple synonyms, none of which agree with the output label; and a dissimilar mnemonic to boot.

2-5. PURPOSE OF FILE GENERATION AND FILE MAINTENANCE:

a. File generation (FG) is used during the initial phase-in for the creation of new files. It is also used at any later time for both the creation of new files and for the structural revision of existing files. As the user's needs evolve and expand he may well find it necessary to create new files or restructure existing files

b. File maintenance (FM) is the routine updating of the data content of existing files. There is a clear distinction between file generation and file maintenance. File generation is primarily concerned with the structure or format of the files while file maintenance is primarily concerned with the data content of the files.

c. Under control of the system monitor either file generation or file maintenance may be called in for execution. The system monitor does this by means of a monitor execute card:

MON$$ EXEQ (name of program). If FG is specified in the monitor EXEQ card the initial phase of file generation is obtained from the system operating file (SOF) for execution. If FM is specified in the monitor EXEQ card the initial phase of file maintenance is obtained for execution.

* (explained in detail later)

Element Name	Element Mnemonic	Size	Type	Valid Entries	Meaning	Remarks	Synonyms	Extract Mode Output Labels
Record Character Cou	RECCT	4	N	Program Maintained	No. of Characters in File Record			CHAR COUNT
Flight	FLITE	30	/	////////Record Control Group (Record ID); see fields below//////////////////				FLIGHT
Airline Name	ANAME	8	A	Name or Std Abbreviation			Airline-Name Airline	AIRLINE
Flight Origin	ORGIN	12	A	City Name	City of Origin	See "LegOrigin" below	Departure-Point Flight-Origin	FLIGHT ORIGIN
Sched. Departure	DEPDT	10		//////////Fixed Group; see fields below//////////////////////			Departure-Date	DEPARTURE DATE TIME
Year	DYEAR	2	N	00-99				YEAR
Month	DMNTH	2	N	00-12				MONTH
Day	DDATE	2	N	00-31				DAY
Hour/Minute	DHOUR	4	N	0000-2400		Last field of groups "FLITE" and "DEPDT"		HOUR
Final Destination	DESTN	12	A	City Name	City of Destination	See "Leg Termination" below	Arrival-Point Destination-City	DESTINATION
Flight Number	FLTNO	3	N	000-999	Airline Flight ID#	Input editing used		FLT NO
Aircraft Type	ACTYP	4	A	Name or abbreviation	A/C Type for Flight		A/C Type Aircraft-type	A/C TYPE
Flight Type	FLTYP	1	A	P - - - C - - - D - - - T - - - X - - -	Passenger Flight Cargo Dead-Head Charter Other, see Remrk	Both Input and Output Conversion Used	Flight-type	FLIGHT TYPE
Capacity	CAPCY	3	N	006-180	No. of Pass. Seats or Tons of Cargo		Capacity	CAPACITY
Periodic Set Control Number	PSCT"n"	8	N	(Program Maintained) 1st Four=No. of Subsets 2d Four=Rel. HOP	Blank Field indicates no entires in Periodic Set "n"	"n" equals periodic set number. (One for each periodic set)		PSCT"n"
Variable Set Control	VSCTL	8	N	Program Maintained 1stFour=No.of Char. 2d Four=Rel. HOP	Blank Field indicates no entries in Variable Set			VSCTL

Figure 2-3. Fixed Field Definition - CMFLA File

Element Name	Element Mnemonic			Valid Entries	Meaning	Remarks	Synonyms	Extract Mode Output Labels
Periodic Subset Sequence Number	PSSQn	3	N	Program Maintained	Indicates Subset Sequence, within Set	"n" is subset sequence number		PSSQn
Leg Origin	LORIG	12	A	Airport or City Name		If airport serves more than one city, or city is served by more than one airport, then use airport name	Leg-Origin	LEG ORIGIN
Leg Termination	LTERM	12	A	Airport or City Name			Leg-Termination	LEG TERMINATION
Geographic Ref.	GEREF	30		//////////Periodic Group: ////////// see fields below//////////		Output Editing Used		LEG COORDINATES
Leg Origin Coord	OCOOR	15	A	Standard * Coordinate Format	Airport Coordinates	Output Editing Used	Leg-Origin-Coord	LEG ORIGIN COORD
Leg Term. Coord	TCOOR	15	A			Last Field of Group	Leg-Term-Coord	LEG TERM COORD
Leg X of Y	LXOFY	2	N	11-99	1st digit is seq no of leg in itinerary; 2d digit is total no of legs in itinerary	Output Editing Used	Leg-X-of-Y	LEG X OF Y
Takeoff Date/Time	TOD/T	6		/////Periodic Group: see fields below//////////			TO-DT/Time Takeoff-Date Time	TO DT/TIME
Takeoff Date	TDATE	2	N	00-31	Day of Month			TO DATE
Takeoff Time	TTIME	4	N	0000-2400	Hour of Takeoff	Last Field of group		TO TIME
Landing Date/ Time	LD/T	6		/////Periodic Group: see fields below//////////			Landing-Date/ Time	LDG DT/TIME
Landing Date	LDATE	2	N	00-31	Day of Month			LDG DATE
Landing Time	LTIME	4	N	0000-2400	Hour of Landing	Last field of group		LDG TIME
Average Altitude	ALTTD	3	N	000-999	Hundreds of feet	Output conversion used	Avg-Alt	AVG ALT
On Board	ONBRD	3	N	000-999	Number of seats occupied or weight of cargo carried		On-Board	ON BOARD
Fuel Loaded at Origin	FUELL	3	N	000-999	Hundreds of pounds	Output conversion used	Fuel-Loaded	FUEL LOADED
Takeoff Gross Weight	GROWT	4	N	0000-9999	Hundreds of pounds	Output conversion used	Takeoff-Weight	TAKEOFF WEIGHT

*NOTE: 15 character coordinate format in the data file cannot be used by OVP operator in Retrieval.

Figure 2-4. Periodic Set One, Field Definition-CMFLA File

Element Name	Element Mnemonic	Size	Type	Valid Entries	Meaning	Remarks	Synonyms	Extract Mode Output Labels
Periodic Subset Sequence Number	PSSQn	3	N	Program Maintained	Indicates Subset Sequence, within Set	"n" is subset number		PSSQn
Location	LOCAN	12	A	Airport or City Name			Location	LOCATION
Unscheduled Maintenance	MAINT	9		Periodic Group; see fields below				UNSCHEDULED MAINTENANCE
Nav/Comm	NCOMM	1	N	In all Fields in Group "MAINT" except "A/C replaced" and			Nav/Comm	NAV/COMM
Airframe	FRAME	1	N	"other" meaning of entries are as follows:			Airframe	AIRFRAME
Electrical System	ELSYS	1	N	Ø - No maintenance performed on this item			Elec-Sys	ELEC SYS
Hydraulic System	HYSYS	1	N	1 - Maintenance performed on this item			Hydr-Sys	HYDR SYS
Engine	ENGIN	1	N	2 - Special case, see REMRK			Engine	ENGINE
Landing Gear	LGEAR	1	N	0 - Same as Ø			Ldg-Gear	LDG GEAR
Environment Control	ENMNT	1	N				Env-Control	ENV/CONTROL
Aircraft Replaced	RPLCD	1	N	Ø - (or 0) 1 - 2 -	Aircraft not replaced Aircraft replaced with same type Aircraft replaced with other type(see REMRK)		A/C-Replaced	A/C REPLACED
Other	OTHER	1	N	Ø - (or 0) 1 -	No entry See REMRK	Last field of group		OTHER
Revenue Loss	RLOSS	9		Periodic Group; see fields below				REVENUE LOSS
Pass. Meals Purchased	MEALS	3	N	000-999	Number of Meals Purchased			MEALS PURCH
Passengers Billeted	ROOMS	3	N	000-999	Number of Billets Provided			PASSG BILLET
Freight Penalty	PNLTY	3	N	000-999	Dollars Freight Penalty	Last field of group Output Editing Used		FREIGHT PENALTY

Figure 2-5. Periodic Set Two, Field Definition - CMFLA File

d. Figure 2-6 shows the generalized make-up of the file generation and file maintenance programs by phase and their place within the 1410 FFS. The remainder of this section provides a general description of each of the phases of file generation and file maintenance. Appendix C provides a general description of the 1410/7010 Operating System for the reader desiring such a description.

2-6. SPECIAL GEOGRAPHIC OPERATOR:

a. It is possible for a user to specify any other polygon-polygon, polygon-circle, or circle-circle relationship by means of his own subroutine. This subroutine, which must use the same format data field and data value as that required for an OVP card, tells retrieval hit or no-hit by means of an appropriate exit from the subroutine. If the data value on the OVP card is a polygon (not a circle) and a special geographic subroutine exists, a subroutine may convert the polygon data value to a special input format for the geographic operator subroutine. This subroutine must be specified in input source 1 of the FIELD card of every field using the special subroutine.

b. Provision must be made for this special operator at file structuring time if it is to be used. A GEOOP card is used in structuring the FFT to identify a special geographic operator subroutine together with a "V" or an "E" operation code. An "E" identifies a subroutine to be used for negative searches. The same subroutine may be specified for both positive and negative searches.

NOTE: A negative search must not be attempted against a cross-indexed field. No special operator (turn) is used at retrieval time. (OVP is still used.) If a file that is structured for the special operator is queried in a multifile query, the general operator will not be available during that run for another file. The special operator subroutine replaced the general operator subroutine for the run so the general will not be available.

2-7. FILE GENERATION:

a. File Structuring.

(1) Assume that the monitor EXEQ card specifies FG. File generation consists of two phases. The phase that is normally executed first in a file generation run is the file structuring (FS) phase*. The purpose of file structuring is to generate a file format table from an input deck containing all of the parameters of the file to be created or changed in structure. This file format table contains a detailed description of the structure (format) of the file. It is used by file maintenance, output processing, retrieval, and the file revision phase of file generation. There must be a file format table for each file.

*When the EXEQ card specifies FG, the file structuring phase is always the initial phase obtained for execution. If the file revision phase is desired without first executing file structuring, a BYPASS FS control card (explained later) can be used to cause bypassing of the file structuring phase so that the file revision phase can be immediately executed.

(2) If file structuring is in the CREATE mode (structuring a new data file) it will:

Create a dummy file tape for eventual use by file maintenance.

Build and output a file format table, based upon the parameters in the input deck.

Return control to the system monitor (without calling in file revision).

(3) If file structuring is in the CHANGE mode (restructuring an existing file) it will:

Build and output a file format table (based upon parameters in the input deck).

Call in file revision (the other phase of file generation) which will then be executed.

b. File Revision. The only way to get the file revision (FR) phase in and executed is for it to be called in by the file structuring phase. The file structuring phase calls in FR after building a file format table in the CHANGE mode or after recognizing a BYPASS FS control card as the first card. Using FS in the CHANGE mode means that the new file format table being built is for an existing file which is presently in a different format as defined by the old file format table. File revision will use both the old and new file format tables to rearrange the existing data in the file from the format specified by the old FFT to the format specified by the new FFT. In addition to the old and new FFT's file revision requires an input deck of control cards and change specification cards. From all of these inputs (and of course the file to be revised) file revision generates the revised data file. After generating the new data file, file revision returns control to the system monitor.

2-8. FILE MAINTENANCE:

a. General.

(1) The file maintenance program consists of six phases. It is capable of updating the data content of any existing FFS file by performing operations (called transactions) on the elements of the file records within the data file. The following file maintenance transactions may be accomplished:

Creation of a new file record and its insertion into a file.

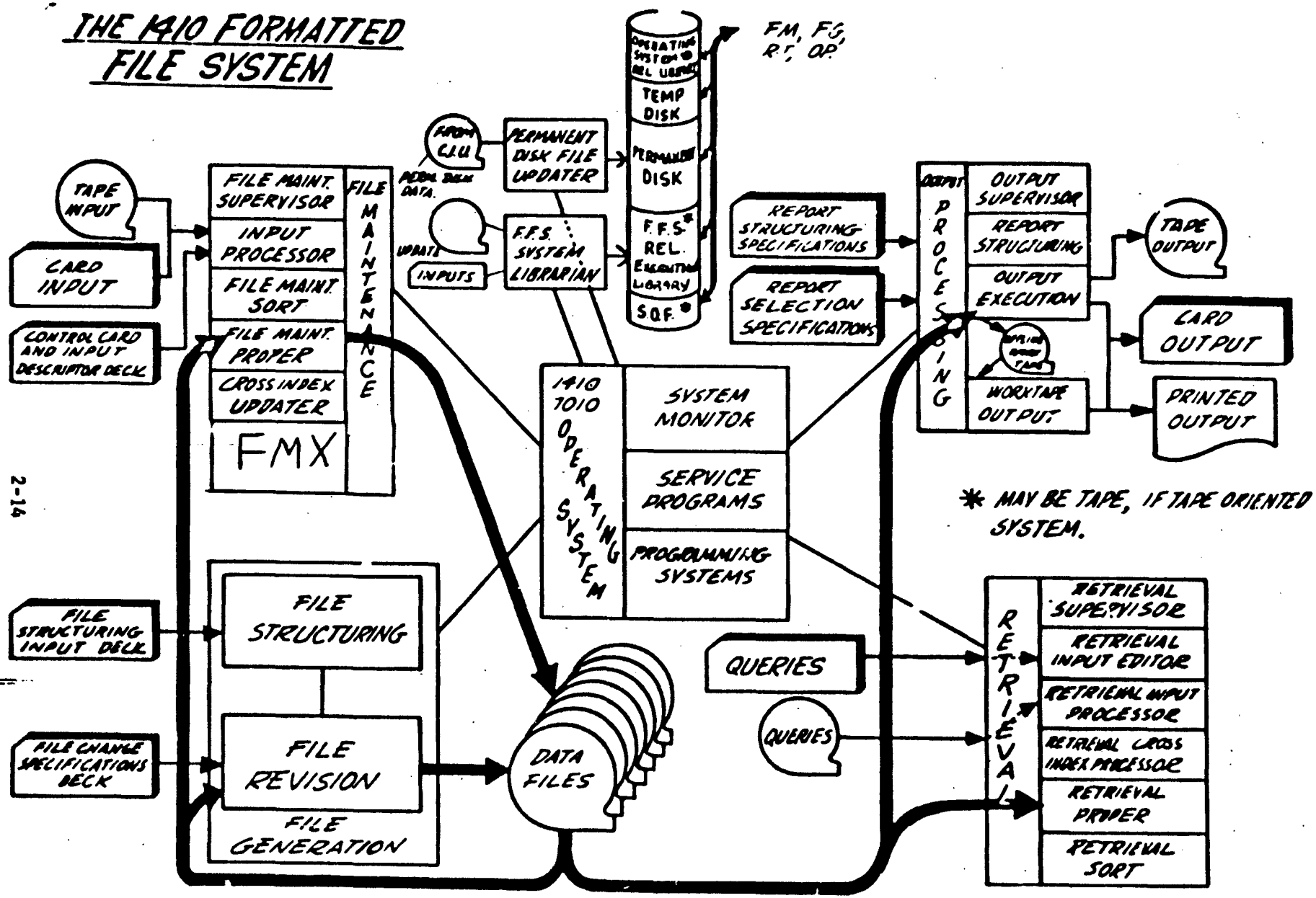


Figure 2-6. The 1410 Formatted File System (Mark I).

Addition of a periodic subset to a periodic set of an existing file record.

Addition of a variable set to an existing file record.

Addition of more unformatted data to an existing variable set of an existing file record.

Changing the data content of a fixed field or periodic field of an existing file record.

Deletion of an entire file record.

Deletion of the data content of a fixed or periodic field or group.

Deletion of an entire periodic set within an existing file record.

Deletion of all periodic subsets having blank data content.

Deletion of a periodic subset within a periodic set of an existing file record.

Deletion of a variable set within an existing file record.

Production of transaction confirmation tape on specified element without affecting data record (NEX operator).

(2) By performing multiple operations, additional tasks may be accomplished. For example, in order to "change" the content of a variable set, it may first be deleted and then the modified data added.

(3) The six phases of the file maintenance program are:

File Maintenance Supervisor

Input Processor (IP)

File Maintenance Sort

File Maintenance Proper

Cross-Index Update (C.I.U.)

FMX (Blank Subset Purge)

b. File Maintenance Supervisor Phase. If FM is specified in the monitor execute card the FM supervisor is called in and executed. Its main functions are to schedule the execution of and provide linkage

between the other phases of FM, and to regain control as each of the FM phases completes. The phase normally called in first by FM supervisor is the input processor (IP) phase.

c. Input Processing Phase.

(1) The purpose of the IP phase is to edit and convert incoming data changes, additions, creations, and deletions into a format compatible with the file to be updated and put this data into a special record form for the FM proper phase. The input conversion subroutines, tables, and editing conventions specified in the FFT are referenced by IP to perform the conversion and editing.

(2) The data input to IP is called the input file. The input file is further broken down into input records. The input file may be in either of two forms:

External Format

Internal Format

(3) External format input is the more powerful of the two types. It allows multiple elements of a file record to be operated upon, for each input record. The arrangement of the data in the input file and its disposition must be specified by an input descriptor deck (IDD). An input control card must accompany the external format input file.

(4) Internal format input follows a fixed internally predefined format. It allows only a single element of a file record to be operated upon for each input record. The disposition of the internal format data, as well as some control information is contained on specific fields of each input record. Internal format is not accompanied by an input descriptor deck. An input control card must accompany each internal format input file.

(5) IP checks the input data description (either internal, or taken from the IDD) against the FFT for the subject file and accordingly edits and converts the input data. After each input record from the input file is read and identified by IP, each element of that record that is to be operated upon is processed individually and output as a single transaction record on the transaction tape. Descriptive data from the FFT is contained in the transaction record in addition to the converted and edited input data, file name, and data source. When the transaction tape is eventually input to the proper phase, the data changes can be made directly to the file without FM proper having to reference the file format table or make any transformation of the data.

d. FM Sort Phase.

(1) The FM sort phase sorts the transaction tape generated by IP. It orders the transaction records by (major to minor):

File Name

Record ID (Control Field/Group)

Set ID (Fixed, Periodic 1-8, Variable)

Periodic Subset Sequence Number

FFT Entry Number (Relative position of the element in its set or subset, as specified in logical record #2 of the FFT.)

(2) This is done to satisfy the input requirements of FM proper, which receives the sorted transaction tape as input.

(3) Since FM sort handles the burden of ordering the transaction tape, IP can process multiple-input files for multiple-data files without regard to the:

Order in which the input files are entered.

Order in which the input data is arranged in an internal format input file.

Order in which the IDD specifies the external format input data is to be processed.

e. File Maintenance Proper Phase.

(1) FM proper generates an updated data file from the sorted transaction tape and the old data file. FM proper uses the transaction records from the sorted transaction tape to make additions or changes to, or deletions from the existing data file. When data is initially input to a new file, the dummy file tape produced by FS is used instead of the old data file. Since the transaction records are sorted into file order, the transaction tape may be essentially merged with the old data file. All control information required to construct the updated file is available from the transaction tape.

(2) In addition to the updated data file FM proper outputs a transaction confirmation tape that is almost like the sorted transaction tape it receives as input. The additional data it contains indicates:

(a) Which of the transaction records were actually used to update the file and which (if any) were not used due to errors.

(b) The data previously in the file which has just been replaced by a change operation or removed by a delete operation.

The transaction confirmation tape may be printed by the output execution phase of output processing. This provides an accurate record of the file maintenance actually performed for review by personnel responsible for the file's data content.

(3) FM proper may output a file data table for the updated data file. The file data table is output for use by the cross-index updater (C.I.U.) phase and is only produced when the file being maintained is cross indexed. Cross-indexed files are usually multireel files. The file data table consists of the record ID (record control group) of the first and last file records on each reel of tape comprising the file. This information may be used by retrieval to prevent having to search every reel of a multireel file for retrievals against that file.

(4) If the file being maintained is cross indexed FM proper also outputs all cross-index transactions onto temporary disk for input to C.I.U. For example, if a record which is referenced in the cross index has just been deleted from the file by FM proper, a cross-index transaction indicating this would be output by FM. C.I.U. would then use this cross-index transaction as a basis for removing the reference to the record in the cross index.

f. FMX - Purge Routine for Subset Deletion. The FMX purge routine is a new phase within the file maintenance group of programs. The intent of the phase is to provide analysts with the optional capability, under the FFS, to delete blank periodic subsets. It is intended to follow an FM proper run which may have generated blank subsets, or to clean up an old data file which may have accumulated blank subsets as a result of periodic field deletions.

g. Detailed Description.

(1) The routine is called via the FM supervisor by means of a control card specifying FMX in the first 3 card columns. An advisory message is given the 1410 operator on the console, specifying tape assignments to be made ready. Tape files are opened and the input data file header label is read. A call is made for the appropriate FFT. If no FFT is found for this file, the operator is advised and given the option of mounting another file or ending the run.

(2) With the FFT in core, the set ID table (LR4) is extracted and moved into a work area. The format of the set ID table is checked and the number of periodic and variable set controls is determined.

(3) One by one the data file records are read, examined, processed, and written onto a new tape reel. If the file record has no periodic sets, it is simply copied without modification.

(4) Periodic subsets are accessed by using available data from the set ID table and each record's set control fields. The number of subsets in a given set is stored, and each one checked for blankness in turn until the stored count is reduced to zero.

(5) When one periodic set of a record has had all its subsets examined, the next set control field is accessed and checking continues as in the above step..

(6) Each time the index register is stepped to another set control field, a corresponding step is made to another register to access the appropriate entry in the set ID table.

(7) Checking continues until all subsets have been examined. If no blank subsets are detected in the record, it is moved to the output area to be written.

(8) If any blank subsets are detected, the record is moved to a work area for purging. This is done in a deletion routine which brings the record from the input area, removes the blank subset, closes the gap in the record, reduces the character count, and updates all set control fields.

(9) When checking is finished a switch is examined to determine whether the record is to be moved to output from the work area or from the input area. When the input data file reaches end-of-file, the two files are closed and unloaded. The FMX Program returns control to FM supervisor, which will return control to the system monitor.

h. Cross-Index Updater Phase.

(1) Need for Cross Indexing a File. If a file is not cross indexed, each term of a retrieval request against the file must be passed against each file record to insure that all file records satisfying the terms of the query are retrieved. Consider the following retrieval against the CMFLA file:

```
IF,DEST∅,EQUALS,ALBUQUERQUE,
AND,DEPDT,IS GREATER OR EQUAL,6412240000,
AND,DEPDT,IS LESS OR EQUAL,6412242400,
AND,ACTYP,EQUALS,DC8B,
AND,FLTYP,EQUALS,P,
```

(2) Without a cross index each file record in CMFLA must be accessed and examined to determine if it satisfies the above logic statements. This serial searching is time consuming and when a large multireel file is involved may result in excessive retrieval time.

(3) The time required to retrieve specific data from a large multireel file may be reduced by cross indexing the file. The necessity of serially searching the entire data file is eliminated by cross indexing. Very basically cross indexing a file consists of keeping a current list of file records, by unique content of the field by which the file is indexed. For example, if the CMFLA file were cross indexed by the Final Destination (DEST∅) field, a list of record ID's of every file record would be maintained by each unique entry in the DEST∅ field as follows:

DEST⍽ = ALBUQUERQUE

AMERICANKENNEDY⍽⍽⍽⍽⍽6411301415

BRANIFF⍽HARTFORD⍽⍽⍽⍽6412241311

List Of Record ID's Of Each Record Having Albuquerque As A Final Destination

BRANIFF⍽HARTFORD⍽⍽⍽⍽6412251514

- - - - - - -

- - - - - - -

- - - - - - -

DEST⍽ = ATLANTA

AMERICANBOSTON⍽⍽⍽⍽⍽6410132300

AMERICANBOSTON⍽⍽⍽⍽⍽6411132400

AMERICANRENO⍽⍽⍽⍽⍽⍽⍽6405140857

List Of Record ID's Of Each Record Having Atlanta As A Final Destination

BRANIFF⍽ALBANY⍽⍽⍽⍽⍽6411251714

BRANIFF⍽BOSTON⍽⍽⍽⍽⍽6411171811

- - - - - - -

- - - - - - -

DEST⍽ = AUSTIN

Etc.

ETC.

Etc.

(4) Now reconsider the retrieval request previously specified against the CMFLA file assuming the file is cross indexed as shown above. The retrieval program will select directly from the cross index a list of all file records (by record ID) which have ALBUQUERQUE in the DEST∅ field, and from the file data table determine which reels of the multi-reel file contain these records. In the example shown, there are several record ID's listed in the cross index for DEST∅ = ALBUQUERQUE, and only the file records having record IDs matching one of those in the cross index will be passed against the logic of the retrieval statement. A savings in time may result from any or all of the following:

Not having to search certain reels of the file at all.

Not having to search the remaining portion of a reel once the file records matching the cross index have been accessed.

Not having to perform all the retrieval logic on each file record accessed.

(5) Eliminating Record ID's in the Cross Index From Consideration Prior to Accessing the File.

(a) When the ALBUQUERQUE parameter of the retrieval statement is passed against the DEST∅ cross index of the CMFLA file three of the record ID's obtained are:

AMERICANKENNEDY∅∅∅∅∅6411301415
BRANIFF∅HARTFORD∅∅∅∅6412241311
BRANIFF∅HARTFORD∅∅∅∅6412251514

(b) Since one of the parameters (DEPDT) of the retrieval statement is an element within the record ID, it is possible to eliminate several of the file records from further consideration by performing retrieval logic on the cross-index list itself. In the example shown, only the second entry in the list would remain as a possible file record which may satisfy the logic, after examination of the DEPDT element. Only the reel containing that file record (as determined by referencing the file data table) would have to be searched, and only the file record having the specified record ID would have the remaining retrieval logic performed on it.

(6) How to Index a File and Specify the Elimination of Fields. When the file is structured by the file structuring phase of file generation, the INDEX card is used to specify the element by which the file is to be cross indexed. A file may have up to two cross indexes, each by a different element. The ELIM card specifies which element in the record ID may be used to eliminate entries in the cross index from consideration prior to accessing the file. These cards are fully explained in Section Three under "File Structuring."

(7) Purpose of the Cross-Index Updater Phase. In order for the cross-index lists to be useful to retrieval they must be kept current. When the data content of an indexed file is updated by a file maintenance

run, the cross index as well as the file data table for that file must also be updated. FM proper outputs all transactions involving cross-indexed fields onto temporary disk along with a new file data table. The cross-index updater phase uses these outputs of FM proper to prepare an "update tape" containing the new cross index and file data table for each cross-indexed file maintained. This tape is then used by the permanent disk file updater program. This program actually updates the cross indexes and file data tables on permanent disk. Retrieval utilizes the cross indexes and file data tables from the permanent disk.

2-9. SECTION TWO SUMMARY

a. A data file is a collection of related information catagorized by subject and/or application. More specifically, an FFS Data File is an ordered collection of similarly formatted data records wherein the data elements have been defined, named, and assigned a relative position. Thus, the 1410 FFS Data Files are also called formatted files.

b. The logical arrangement of data elements that is required to describe an activity, event, objective, person, place, thing, etc., is called a file record or a data record.

c. The file format specifies the pattern or structure of the file records, as well as determining the ordering of the file records within the file.

d. The data elements in a file record whose values do not change during the span of space or time covered by the file record are called fixed data.

e. Periodic data is the data elements which may assume several different values over the span covered by the file record.

f. Data that cannot be conveniently classified as either fixed or periodic but which can be catagorized as related remarks or comments is called variable data.

g. The following logical elements of data are the components of a file record.

Field - the smallest defined logical element of data that can be processed or manipulated by the FFS. It may consist of 1 or more adjacent character(s).

Fixed field - a field that can appear only once within a file record. Contains fixed data.

Periodic field - a field that may appear more than once within a file record. Contains periodic data.

Group - multiple adjacent fields of the same set that are related and have been defined as an entity. Groups are processed and manipulated as are fields. Depending upon the type of fields grouped, a group may be periodic or fixed.

Periodic subset - one or more uniquely defined periodic fields/groups which are so related that they must be repeated as a unit in the file record.

Periodic set - a collection of one or more identically structured periodic subsets. Up to 8 periodic sets are allowed in a file record.

Fixed set - a collection of all the fixed fields in a file record, including the fixed fields which are program maintained. The first set in the file record.

Variable set - a single, nonrepeating, unformatted, variable length field of data which appears last in the file record.

h. Groups may be contained within other groups.

i. The record ID (record control group) consists of one or more of the initial fixed data fields of the file record. The data content of these fields gives each file record a unique identity, as well as being the sort key for sequencing the file records within the file.

j. Sample File CMFLA - Commercial Flights File.

Fixed set is 11 fixed data fields, 2 groups, and 4 fixed program maintained fields.

Record ID is 30 characters long, consisting of six fields, four of which are grouped.

Periodic set 1 has one subset for each leg of the flight; therefore, it will always have at least one subset. Each subset has 13 fields, 3 groups, and a three digit PSSQ1 field.

Periodic set 2 may have no entry or multiple entries since a subset exists only for each location at which unscheduled maintenance occurs. Each subset has 13 fields, two groups, and a three digit PSSQ2 field.

Variable set may contain any remarks desired, or none at all.

Figures 2-2, 2-3, 2-4, and 2-5 provide the best summary of the CMFLA sample file.

k. File generation (FG) is used to structure new files, and to revise the structure of existing files. File maintenance (FM) is used to put new data in files, update existing data in the files, and to delete unwanted data in the files.

l. File generation consists of the file structuring (FS) and file revision (FR) programs.

m. File structuring generates a file format table (FFT) from an input deck defining all of the parameters of the file whose format is to be created or changed. The FFT output by FS must be put on the FFS Relocatable Execution Library by the FFS Librarian program.

n. The detailed format (structure) of a file is contained in that file's FFT. The FFT for a file is referenced by the various FFS programs as required.

o. File revision is used to rearrange the existing data in the file from one format to another. After FS has generated a new FFT for the file to be revised, FR compares the old and new FFT's and uses the supplied change cards to determine how to rearrange the data in the old file so that it corresponds to the format specified in the new FFT. Then FR performs this rearrangement.

p. File maintenance consists of six phases, which are:

FM supervisor schedules the other FM phases and provides linkage between them.

Input processor gets the external or internal format input file; extracts from it the input data and checks it; and edits and converts the input data additions, creations, changes or deletions into a format compatible with the file to be maintained. Combines the data transactions with control information and then puts the transactions out as "transaction records" which are compatible with FM proper.

FM sort sorts the transaction records into the order of the file to be maintained to make the actual maintenance process faster and easier.

FM proper generates an updated data file from the sorted transaction records and the old data file.

Cross-index updater is executed if the file being maintained is cross indexed. It prepares an "update tape" for the file's cross-index table and file data table. Permanent disk updater uses this tape to actually perform the updating of the tables.

FMX this phase will delete all blank periodic subsets from the FFS data file.

q. Cross indexing a file consists of keeping a current list of file records by the unique data content of the element of the file record by which the file is cross indexed. This is called the cross-index table. If the file is cross indexed twice, then it has two cross-index tables. A file data table is also kept for multireel cross-indexed files. It indicates exactly what segment of the file is on each reel. Whenever an element by which a file is cross indexed is used as a retrieval parameter, retrieval can save file search time by using the cross-index table(s) and file data table for the file.

r. Special Geographic Operator is a subroutine which takes the place of the general geographic operator at file retrieval. It is employed if specified in the FFT by a GEOOP card at file structuring time. This special geographic operator is called for by use of one of the geographic operator words in a query. This area will be covered in the retrieval and output processing book.

SECTION THREE

FILE GENERATION

3-1. FILE STRUCTURING:

a. General.

(1) The initial phase of file generation is called file structuring (FS). File structuring requires as input a deck of cards containing the parameters which define the format of the file. These parameters are used to develop a table which provides the complete definition of the file's format to the rest of the 1410 FFS. This table, called the file format table (FFT), must be developed for each data file associated with the system. The FFT contains not only the specifications for the format of the file, but also partial specifications of external formats and conversions required between the external formats and the file format.

(2) The file designer must be aware of the user's information handling objectives in order to properly construct a data file. He must be concerned with the manner in which each logical element of data is utilized throughout the system. Each of these logical elements of data (e.g.; field, group, set, periodic subset) must be evaluated in terms of how it is entered via file maintenance, queried by retrieval and presented by output processing.

(3) Briefly, some of the parameters in the input deck to FS which define the format of the file include the:

File ID (file name).

Record ID (record control group).

Subroutines and tables used for input and output conversions for fields or groups.

Edit control words for the input and output of fields or groups.

Field names, order of appearance, and set membership.

Field sizes, and whether alphameric* or numeric**.

Group names with list of all fields included within the group.

Variable set name and number of characters to be output (by extract mode) per line.

*alphameric - left justified, padded with blanks.

**numeric - right justified, padded with zeros.

Input source ID for different elements of data.

Element(s) by which the file is cross indexed.

Element by which entries in the cross-index table may be eliminated from consideration for retrieval purposes.

Output labels for extract mode.

Type of logic mode for retrieval to use for data in each periodic set.

If special geographic operator is to be used for the file.

(4) File structuring uses these specifications to build a file format table. If the FS job is in the CREATE Mode (i.e., a new files format is being created), the file structuring program builds the file format table and generates a dummy file tape for later use by file maintenance. File structuring then returns control to the system monitor. In the CHANGE Mode (i.e., the structure of an existing file is being changed), the new file format table is structured as before, no dummy file tape is generated, and file structuring then calls in the file revision (FR) phase of file generation. File revision will use both the old and new file format tables (as well as specifications from change cards) to revise the existing data file into a "new" data file containing no new data but having a modified format.

(5) Output from file structuring consists of:

A listing of the input deck with messages noting any diagnosed errors.

The file format table which is:

(a) Listed on the printer.

(b) Written in relocatable form on tape, for later input to the system librarian, which puts the new FFT on the FFS Relocatable Execution Library for use by other 1410 FFS programs.

(c) Punched in relocatable form on cards, which may be used for the same purpose as the FFT on tape. (Provides backup.)

If CREATE mode is used, a dummy file tape is produced to be used later as input to file maintenance (FM proper phase).

(6) Figure 3-1 illustrates the overall operation of file structuring showing inputs, outputs, and the major operations performed.

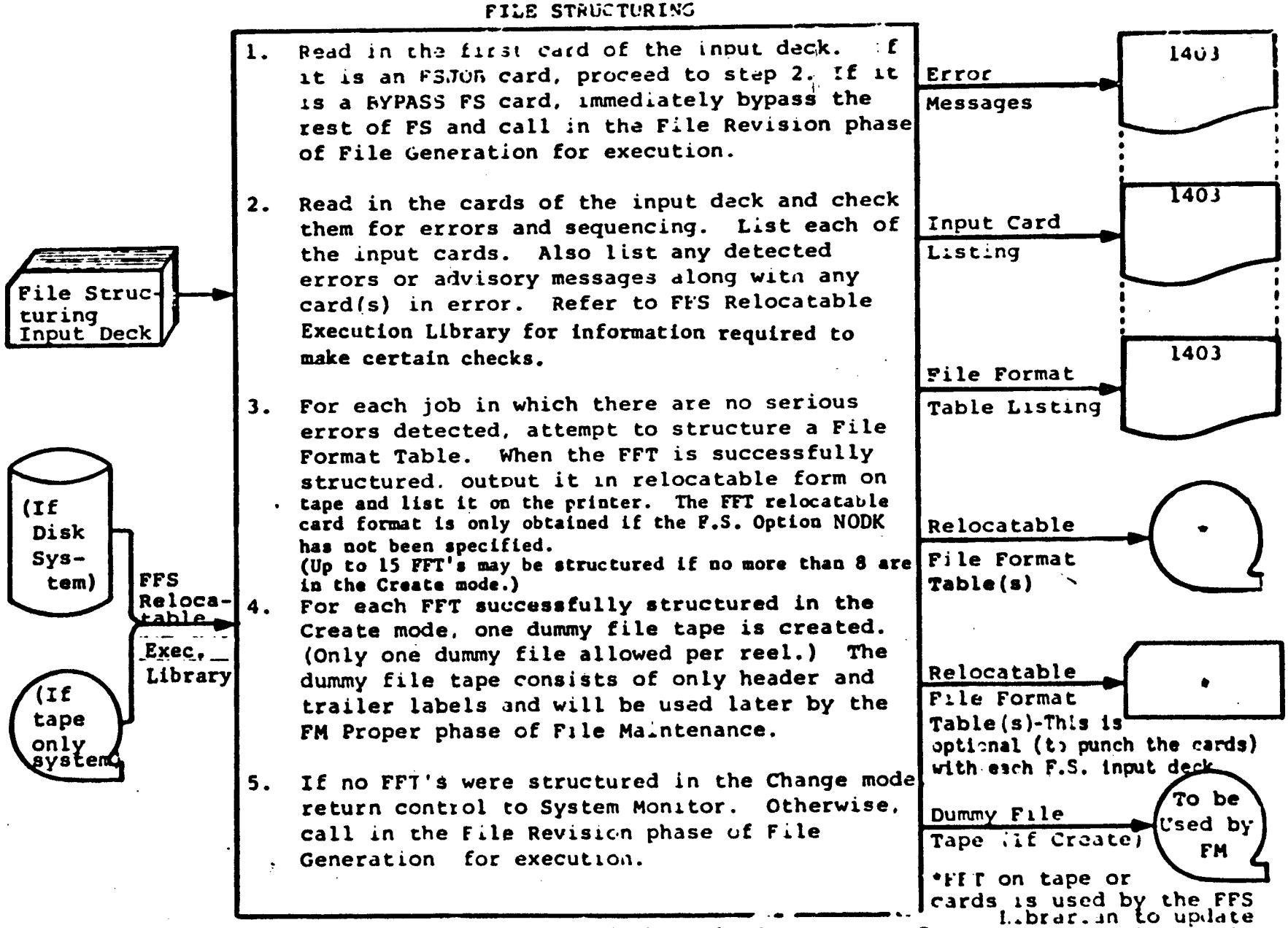


Figure 3-1. General Operation of the File Structuring Phase

b. Quantitative Limits.

(1) One of the general considerations in the design of a data file is the quantitative limits placed on the various data elements. These limits have been set on the basis of an analysis of the anticipated information handling environment, in conjunction with considerations of the characteristics of the 1410 Data Processing System. The following list gives some of the more important limitations imposed upon the size of data elements in the IDHS 1410 FFS.

Maximum number of fields/groups that can be defined within a file record within any file is 299.

(2) Although it is theoretically possible to structure an FFT that contains 299 fields/groups, other FFS Programs impose an 11,000 character size limitation which prohibits the FFT from attaining the theoretical maximum. The size of any FFT depends primarily on the number of fields/groups but also varies according to the following seven elements:

Number of sets

Number of edit fields

Number of input sources

Number of input subroutines

Number of output labels

Size of output labels

Size of edit fields

(3) Using a formula developed for computing the size of an FFT (see appendix E) and assuming that the above 7 elements are average (that which is required for normal applications) the 11,000 character limitation will allow only 190 fields/groups to be structured.

(4) In order to increase the number of fields/groups some of the above 7 elements can be decreased in size. For example, assume that output labels can be reduced to an average of 2 characters per field/group, and the input source description can be reduced to an average of 4 characters per field/group. Approximately 271 fields/groups can be structured within the 11,000 character limitation.

Maximum number of periodic sets that can be defined within a file record of any file is 8.

Maximum number of periodic subsets within any given periodic set is 599.

Maximum number of characters that can comprise the record ID (record control group) is 30.

Maximum number of characters comprising any field or group used as:

a data value (parameter) for retrieval is 56;
a conditional value (parameter for output processing is 52.

Maximum number of characters comprising any field or group not used as specified above is 910.

Maximum number of characters in any data record is 5400.

(5) It is important to add that the 1410 FFS has been written in a manner which allows several of these limits to be altered without the necessity of significant reprogramming (assuming the computer configuration can accommodate the change; e.g., has sufficient core storage available for an increase in the size of some item). It is also important to note that a file design that initially exceeds one or more of the limitations may usually be implemented without significantly changing the intent or capabilities of the original design. Where the maximum record size limitation is initially exceeded, the file design might be accomplished by breaking the record into two entries, based upon the logical break represented by the periodic sets. Where the maximum number of field/group is initially exceeded, the data from several separate fields may be consolidated into one field. The edit feature of output processing may then be used to reconstitute the data into separate entities at output time.

c. Data Element Placement. The file designer must select the optimum placement of each data element within the file record, the grouping of related elements, conversions required,and the ordering of the file records within the file. The latter is determined by the selection of the fields which comprise the record ID (record control group). In addition to determining the sort order of the file, the record ID provides uniqueness to each file record. As previously discussed, the characteristics of every file record have unique data content in its record ID. The nature of the individual data elements will largely determine whether they are placed in the fixed set or in a periodic set. The determination of the number of periodic sets and the control of each must be made by the file designer, based upon the interrelationships and groupings of the periodic data, as well as its intended use.

d. Variable Set Data. The nature of some information precludes the ability to and/or the desirability of formatting it. This qualitative type of information may be termed "Remarks," "Comments," or "Description." Such information may be placed in the variable set. The variable set is treated as a single variable length field of data by the system. From the user's viewpoint, the major disadvantage to this type of data is that

the system must treat it as unformatted, thus preventing the performing of retrieval logic on the data content of the variable set. In addition, file maintenance and output processing must also handle the variable data in "blocks" which tend to be unwieldy. Thus, the decision to place data in the variable set should be carefully weighed, because it takes as much effort (i.e., input preparation, keypunching, etc.) to enter this data as it does to enter formatted data, yet it cannot be used as a parameter for retrieval nor can portions of it be selectively printed; and, when it is included in output reports, its variability makes it difficult to manipulate by standard procedures.

e. FS Input Card Specifications. There are thirteen types of input cards acceptable to FS:

FSJOB	SUB	GROUP	INDEX
ENDFS	TAB	FIELD	ELIM
BYPASS	EDIT	VSET	GEOOP
			*(comment)

(1) Common Elements.

(a) Except for the *(comment) and BYPASS cards, all cards contain the card type identification in columns 8 through 12. All cards except FSJOB, ENDFS, GEOOP, and *(comment) cards have a mnemonic or label in columns 2 through 6. If a deck is not sequence numbered, columns 73 through 80 should be blank. If a deck is sequence numbered, columns 73 through 75 should contain an ascending (by any increment) sequence number and 76 through 80 should contain a deck ID.

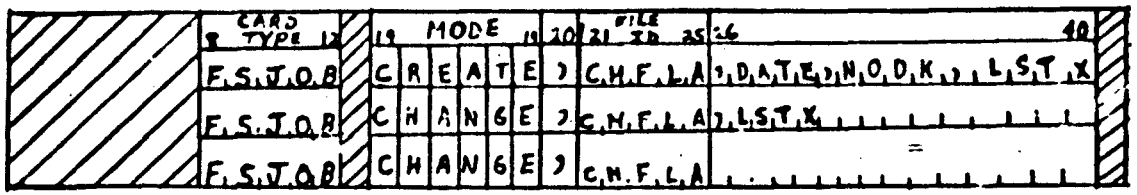
(b) Continuations of all cards except the *(comments), INDEX, ELIM, and GEOOP cards are effected by repeating columns 1 through 13 on the continuation card(s). Up to three continuation cards are allowed. Column 14 of the continuation card takes up where column 72 of the preceding card left off.

(c) In addition to using the *(comment) card for remarks, any remarks desired may be placed after the last normal entry in a card except INDEX, ELIM, and GEOOP. One or more blanks must be left between the last normal entry and the beginning of the remark. The remark must not extend beyond column 72 of the card.

(d) When FS is used to structure the FFT's for more than one file, the groups of input cards for each of the various files MUST BE SEQUENCED IN ASCENDING ALPHABETIC ORDER BY FILE ID.

(2) Detailed Input Card Formats. The formats and allowable entries for each FS input card are given by card type, in figures 3-2 through 3-9. Figures 3-11A and 3-11B give the allowable sequence of cards in a job.

"FSJOB" Card



This card signals the beginning of a group of input cards containing the data for constructing an FFT for the file specified in columns 21-25. It also specifies whether the CREATE or CHANGE mode is to be used during the structuring of the FFT. There must be one FSJOB card for each FFT to be structured by FS.

Cols 01-07 = Blanks

Cols 08-12 = FSJOB

Col 13 = Blank

Cols 14-19 = CREATE or CHANGE. CREATE causes the production of a dummy file tape for file maintenance. If the FFT for the specified file is already in the FFS Relocatable Execution Library, an advisory message is given. CHANGE serves as a positive check on the File ID, since it requires that a FFT for the specified file already be in the FFS Relocatable Execution Library. You receive an error message if the old FFT is not found at file revision time.

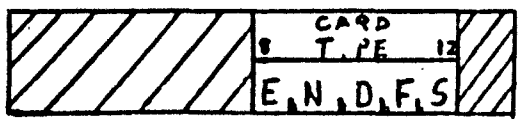
Col 20 = Comma

Cols 21-25 = File ID. File names must be unique within the system. The fifth character must be an A. The first four positions may contain from two to four left justified characters with no embedded blanks. Thus AIRbA or CIbbA are legal but AbIRA, AIRAb, AbbbA and bAIRA are not.

Cols 26-40 = File Structuring Options. The options may be in any sequence. Omitted parameters are not represented by commas. The DATE option will cause CREATE and CHANGE date fields to be inserted into the FFT immediately following the control fields. The CREATE date field is called "CREDT"

Figure 3-2. FSJOB and ENDFS Cards

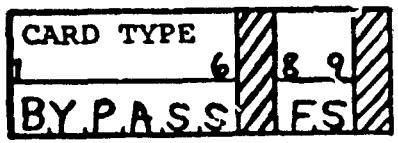
and the change date field is called "CHGDT," with each field being six characters in length. The six-character field is effectively subdivided into three two-character fields which represent the year, month, and day respectively. NODK indicates that the punching of the FFT deck is to be inhibited. LSTX provides the option for extra listings of the FFT. The "X" in the LSTX must be numeric from 1 through 9 and indicates the number of extra copies desired.



This card signals the end of all of the input cards in the FS input deck. It causes FS to either return control to the system monitor or to call in file revision for execution, depending upon the mode (CHANGE or CREATE). There must be only one ENDFS card per input deck, regardless of the number of FFTs being structured.

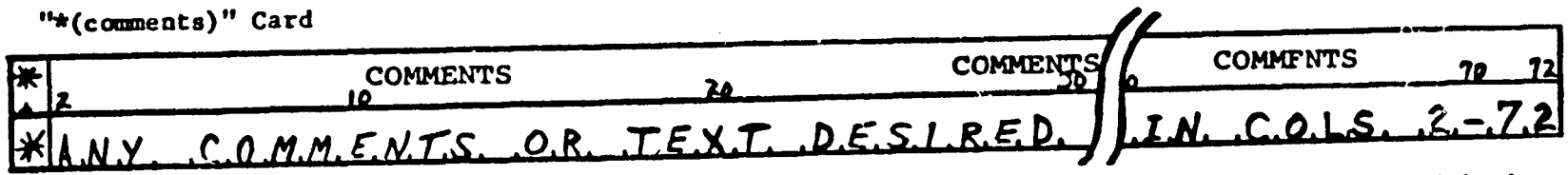
Figure 3-2. (Continued), FSJOB and ENDFS Cards.

"BYPASS" Card



This card causes file structuring to bypass all of its normal operations and immediately call in file revision for execution. If used, this card must immediately follow the "MON$$ EXEQ FC" card (an FSJOB card is not allowed in this case).

"*(comments)" Card



This card has no effect on the structuring of the FFT. It is listed on the printer along with the rest of the input card listing. The file designer may use as few or as many of these cards, in any part of the input deck, as he desires.

Figure 3-3. BYPASS and *(comments) Cards.

f. Editing.

(1) The 1410 FFS provides for the automatic input and output editing of file data. Editing may be used to cause the insertion of most 1410 system characters:

Between data characters.

As a prefix to data characters.

As a suffix to data characters.

Any combination of the above three insertions.

In addition to the above insertions, editing may be used to:

Suppress zeros to the left of significant digits.

Selective insertion of the credit symbol, comma, minus sign, asterisk, dollar sign, and decimal.

(2) The edit control word (supplied by the EDIT card) specifies how the data field is to be edited. It specifies which characters are to be inserted, how they are to be inserted, and where zero suppression is to occur.

(3) The edit control word is divided into two parts: the body (used for punctuating the data field) and the status portion (containing the special characters).

(4) The body of the EDIT control word is that portion beginning with the right-most blank or zero and continuing to the left until the data field ends. The remaining portion (left and/or right) of the control field is the status portion. The edit control word (as supplied by the EDIT card) is those characters between the first and the last @ symbols. All numerical, alphabetic, and special characters can be used in the edit control word. However, some of these have special meanings, as listed in the table.

Edit Control Word Character	Function
ß (blank)	Replaced with the character from the corresponding position of the data field.
0 (zero)	Used for zero suppression. Replaced with a corresponding character from the data field. The rightmost 0 in the edit control word indicates the rightmost limit of zero suppression.
. (decimal point)	Undisturbed insertion in the edited data field in the position where written, unless decimal control was in effect, and the data field did not contain a significant digit. (See Decimal Control.)
, (comma)	Undisturbed insertion in the edited data field in the position where written, unless zero suppression takes place and no significant numerical character is found at the left of the comma.
CR (credit) C R	Body portion: undisturbed insertion in the position where written. Status portion: if sign of the data field is plus, these positions are replaced by blanks. If the sign of the data field is minus they are inserted undisturbed in the edited data field in the positions where written. (Also see sign control.)
- (minus)	Same as CR.
& (ampersand)	Causes a blank space in the edited data field. It can be used in multiples.

Edit Control Word Character	Function
* (asterisk)	Status Portion: undisturbed insertion in the position where written. Body Portion: (See asterisk protection).
$ (dollar sign)	Status Portion: undisturbed insertion in the position where written. Body Portion: (See floating dollar sign).

insertion

EXAMPLES OF INSERTION	
Data Field	65
Edit Control Word	@19ßß@
Edited Data	1965
Data Field	431012N0712813W
Edit Control Word	@ßßDEGßßMINßßSEC&ß&,ßßßDEGßßMINßßSEC&ß@
Edited Data*	43DEG10MIN12SEC N.,071DEG28MIN13SEC 6
Data Field	2615
Edit Control Word	@ßß&@&ßß@
Edited Data	26 @ 15

* Since edit ng is intended for numeric data, the zone bits of the low rder position are removed. For this reason, the W becomes a 6.

(5) Zero Suppression. Zero suppression is the replacement of unwanted zeros to the left of significant digits with blanks. To accomplish this a zero is placed (in the body of the edit control word) at the desired right-most limit of zero suppression.

EXAMPLES	
Data Field	0010900
Edit Control Word	@$ƀƀ,ƀƀ0.ƀƀ@
Edited Data	$ 109.00
Data Field	00000596
Edit Control Word	@ƀƀ0ƀƀ.ƀƀ@
Edited Data	05.96
Data Field	0012500045
Edit Control Word	@ƀƀƀƀƀ&-&ƀƀ0ƀƀ@
Edited Data	125 - 00045

(3) Asterisk Protection. This type of editing is used when it is necessary to have asterisks appear to the left of significant digits. The edit control word is written with the asterisk in the body part, to the left of the zero suppression code.

NOTE: Asterisk protection and floating dollar sign cannot both be used in the same edit control word.

EXAMPLES	
Data Field	00257426
Edit Control Word	@$ƀƀƀ,ƀ*0.ƀƀ@
Edited Data	$**2,574.26
Data Field	00007426
Edit Control Word	@$ƀƀƀ,ƀ*0.ƀƀ@
Edited Data	$*****74.26

(7) Floating Dollar Sign. This feature causes the insertion of a dollar sign in the position at the left of the first significant digit in an amount. The edit control word is written with the $ in the body to the left of the zero suppression code.

NOTE: Floating dollar sign and asterisk protection cannot both be used in the same edit control word.

EXAMPLE	
Data Field	00257426
Edit Control Word	@ßßß,ß$0.ßß@
Edited Data	$2,574.26

(8) Sign Control. The symbols CR or - can be placed at the left or right of a negative field. The edit control word is written with the CR or - symbol in the status portion. (Due to a peculiarity of the 1410, C's and R's individually have the properties attributed to CR.)

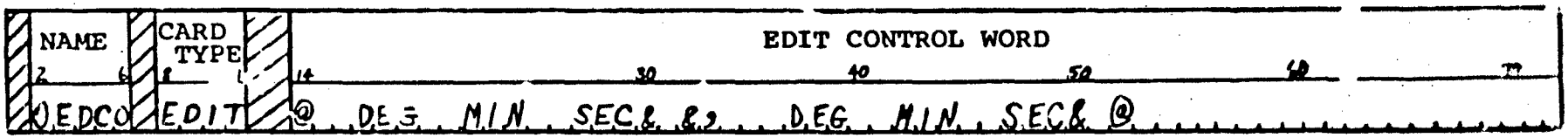
EXAMPLES	
Data Field	0037894M
Edit Control Word	@CR&ßßß,ßß0.ßß@
Edited Data	CR 3,789.44
Data Field	00008940
Edit Control Word	@-&ßßß,ßß0.ßß@
Edited Data	89.40
Data Field	0025742N
Edit Control Word	@$ßßß,ßß0.ßß&CR@
Edited Data	$ 2,574.25 CR
Data Field	0000894L
Edit Control Word	@-&$ßßß,ßß0.ßß@
Edited Data	- $ 89.43

(9) Decimal Control. This editing feature insures that decimal points print only when there are significant digits in the data field. The edit control word is written with a decimal point in the body to the left of the zero suppression code.

EXAMPLES	
Data Field	00000
Edit Control Word	@ƀƀƀ.ƀ0@
Edited Data	(blank)
Data Field	29437
Edit Control Word	@ƀƀƀ.ƀ0@
Edited Data	294.37
Data Field	00001
Edit Control Word	@ƀƀƀ.ƀ0@
Edited Data	.01
Data Field	00101
Edit Control Word	@ƀƀƀ.ƀ0@
Edited Data	1.01

The reader desiring more detailed information concerning editing is referred to the "IBM 1410 Principles of Operation" manual, form number A22-0525.

"EDIT" Card



DIAM 65-9-1

Col 01 = Blank

Cols 02-06 = Two to five character left justified edit control word name. The fifth character cannot be an "S". If only one field (or group) uses this particular edit control word, use the name of that field (or group) here for the edit control word name If more than one field/group uses this edit control word, assign a unique name (by file) to the edit control word.

Col 07 = Blank

Cols 08-11 = EDIT

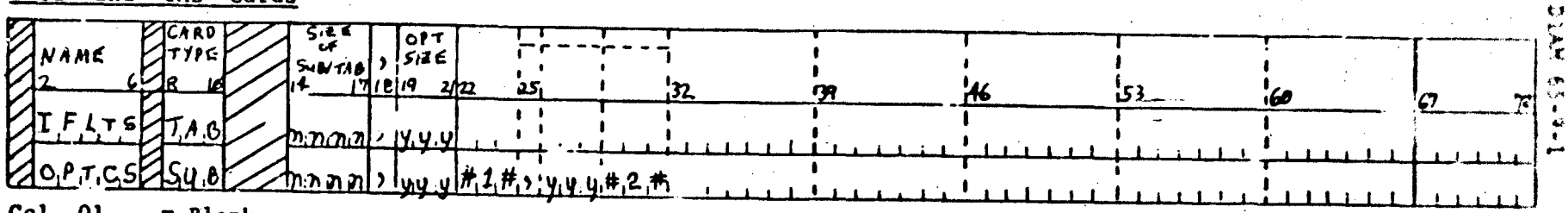
Cols 12-13 = Blanks

Cols 14-XX = The edit control word, which consists of those characters between the first and last @ symbols. By using continuation cards the edit control word may be as long as 132 characters.

Figure 3-4. EDIT Card.

g. Subroutine Verification. The file structuring program has been designed to make many checks on the specifications supplied to implement a file. Among these is a check on program areas used for input/output conversion subroutines and tables, and a check for their presence on the FFS Relocatable Execution Library. The SUB and TAB cards allow the file designer to describe these input/output conversion subroutines and tables so that such checks can be made. The names of the subroutines and tables will also appear in the FIELD and/or GROUP cards that define the fields/groups utilizing the conversion subroutines and tables. (It should be noted that an element comprising all or part of the record ID (Record Control Group) cannot be processed by an input conversion subroutine.)

"SUB" and "TAB" Cards



Col 01 = Blank.

Cols 02-06 = The name of the subroutine or table, consisting of five non-blank characters. The fifth character must be an "S". Subroutine and table names must be unique, by SYSTEM.

Col 07 = Blank.

Cols 08-10 = TAB or SUB. The system makes no distinction between SUB and TAB entries. The provision for either of two entries is only for the convenience of the file designer. If the conversion subroutine is of the table lookup type, the file designer may wish to use TAB. If not a table lookup conversion subroutine he may use SUB. (Or he may just use SUB for any conversion subroutine.)

Cols 11-13 = Blanks.

Cols 14-17 = The (four digit) number of characters of core storage that constitute the entire subroutine or table. If the subroutine is not in the FFS Relocatable Execution Library or the size specified is less than 60 or greater than 9999, an advisory message will be printed out.

Col 18 = Comma.

Cols 19-21 = A three-digit number indicating the number of characters which are present in the output of the subroutine or table. Normally subroutines and tables provide only a single output size. However, there are cases where more than one output size can be produced (e.g.; conversion subroutines that handle more than one field).

Cols 22-24 = In cases where multiple output sizes are produced, #1# is placed immediately following the "first" output size.

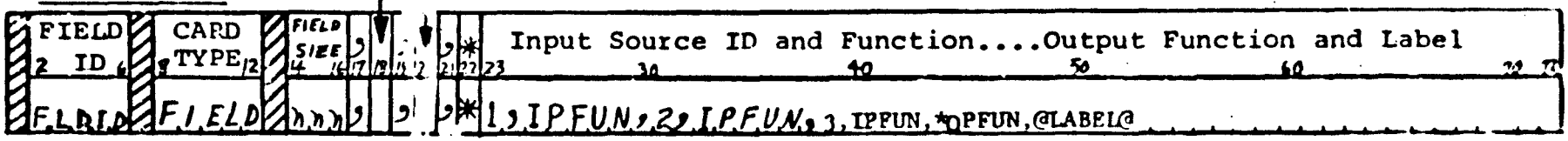
Cols 25-31 = The "second" output size entry consisting of a comma, a three digit output size and #2#.

Cols 32-XX = The other output size entries (up to a maximum of nine). Each entry follows the format of cols 25-31.

Figure 3-5. SUB and TAB Cards.

"FIELD" Card

Field Type ID
Logic ID



Col 01 = Blank.
Cols 02-06 = A two to five alphanumeric character field name, left justified, with no embedded blanks. The field ID must be unique within a given file. (See note page 3-22.)
Col 07 = Blank.
Cols 08-12 = FIELD
Col 13 = Blank.
Cols 14-16 = A three-digit number indicating the number of characters in the field. The size of a field may be from 001 to 910 characters, depending upon its use, and subject to limitations. A field used as a data value (parameter) for retrieval cannot exceed 56 characters in size. A field used as a conditional value (parameter) for output processing cannot exceed 52 characters.
Col 17 = Comma.
Col 18 = Field type and set membership, as follows:
- C = Control Field (A fixed field which is a component of the record ID)
- X = Ordinary fixed field.
- 1 = Periodic field in periodic set 1.
- 2 through 9 = Periodic field in periodic set 2 through 8, respectively.

Col 19 = Comma.
Col 20 = The "built-in" logic mode for use by retrieval (and output) when handling periodic data. (Refer to the "Retrieval and Output" manual for a detailed explanation of the significance and appropriate use of each of the logic modes.) This column should be left blank for all fixed fields. All of the fields of a given periodic set must have the same type of logic mode specified for them.
- I = Inter mode
- S = Scan mode
- N = Non-scan mode
- T = Terminate mode

} Three forms of the "INTRA" mode

Figure 3-6. FIELD Card.

Col 21 = Comma.
Col 22 = Asterisk.
Cols 23-XX = "Input Source ID and Function." There may be from one to nine input source IDs and functions. The file designer must consider whether the input data to the field is in numeric or alphameric format, or whether it should be edited or operated upon by a subroutine or table. All input sources must be considered. ID #1 is the retrieval query card. ID #2 is the internal format file maintenance input card. IDs #3 through #9 are for external format input to file maintenance. If, after considering all sources, all of the data for the field is to be treated alike (i.e., use the same input function), then the source ID in column 23 should be a $, followed by a comma, followed by the input function. (e.g.: ALPHA, NUMER, edit name, or subroutine name) If any one or more input sources of data to the field is treated differently than the other sources, then the source ID in column 23 should be a 1, followed by a comma, followed by the input function for source number one. Follow the input function entry with a comma, and then enter a 2, followed by a comma, followed by the input function for source number two, followed by another comma. In a similar manner enter the source ID and input function for source number three and for any additional sources that will be used. (Succeeding source IDs must be in sequence.) The input function may be any of the following:

(a) ALPHA - Alphanumeric (left justified, padded with blanks)
(b) NUMER - Numeric (right justified, padded with zeros)
(c) The NAME of the applicable edit word.
(d) The NAME of the applicable subroutine or table. If the subroutine or table specified has a multiple output size function, it will have subscripts in the form #n#. In this card the subroutine NAME must be followed by the applicable subscript (in the form #n#) if the specified subroutine or table is subscripted.

Col XX+1 = An asterisk to separate the "input source ID and function" entry from the "output function" entry.
Cols XX+2-YY = "Output function." Enter the output function to be used for the EXTRACT mode of output processing. The output function may be any of the following:

(a) Five blanks, meaning the data is to be output "as is" in file format.
(b) The NAME of the applicable edit word.
(c) The NAME of the applicable subroutine or table. This name must be subscripted in the form #n#, if the specified subroutine or table is so subscripted.

Figure 3-6. (Continued), FIELD Card.

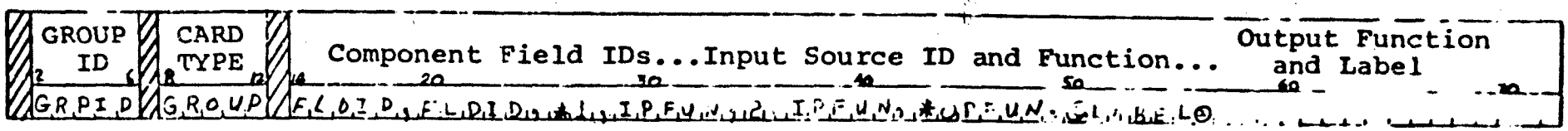
Col YY+1 = Comma.

Cols YY+2-ZZ = "Label." The label to be used for the EXTRACT mode of output processing. The label consists of those characters between @ symbols and may be from zero to 56 characters in length. The size of the label is not affected by the size of the data field.

Note: The mnemonics for fields and groups to be used in retrieval may not consist of two or more of the leading characters of any retrieval operator word. For example: EQUALS is a retrieval operator word. Therefore EQ, EQU, EQUA, and EQUAL are not valid field/group mnemonics. But EQXXX is a legal mnemonic since the name as a whole is different from a truncation of the word EQUALS. For fields/groups used in output, the following mnemonics are invalid: COUNT, LOGNO, REQMO, SORT, TOTAL. Output operations are also illegal field/group mnemonics.

Figure 3-6. (Continued), FIELD Card.

"GROUP" Card



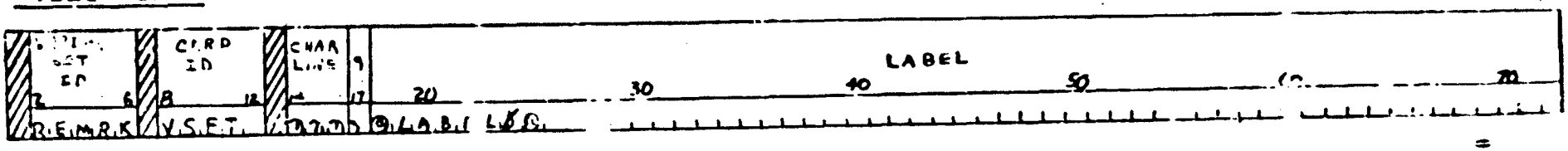
Col 01 = Blank.
Cols 02-06 = A two to five alphanumeric character group name, left justified, with no embedded blanks. The group ID must be unique within a given file. (See note page 3-22.)
Col 07 = Blank.
Cols 08-12 = GROUP
Col 13 = Blank.
Cols 14-XX = Enter the ID's of the fields to be grouped within this group. From one to 20 field ID's may be entered. They must be listed in file sequence, with no omissions between the first and the last (i.e.: only adjacent fields may be grouped). The last field listed to be grouped, must be the last field which was defined. That is, the GROUP card must immediately follow the FIELD cards which define the fields to be grouped. The one exception to this rule is that the group ID of a previously defined group may be substituted for its corresponding field ID's, when that group is to be wholly contained within the group defined by this GROUP card. In order for an entire periodic subset to be manipulated (e.g.: deleted by FM), be careful to group together all of the field/groups of any one periodic set (including the PSSQn field), and to assign it a group ID. For example, after defining the last element of periodic set two, a GROUP card should appear listing the first field to group as PSSQ2, and continuing with all of the other fields/groups in the set, through the last. A logical group ID for this group might be "PSET2," or an ID describing the data content of a subset within the set might be more useful. Only when the first field specified in a GROUP card is PSSQn, is there a subset entry (#5 Type ID) made into L.R. number 2 of the FFT.

Col XX+1 = Asterisk

Subsequent Columns = From this point on the coding continues as from the first asterisk (in col. 22) in the FIELD card. (Input source ID and function - - - output function and label.)

Figure 3-7 GROUP Card.

"VSET" CARD



Col 01 = Blank.
Cols 02-06 = A two to five character name for the variable set. The name should contain no embedded blanks and should be left justified.
Col 07 = Blank.
Cols 08-11 = VSET
Cols 12-13 = Blanks.
Cols 14-16 = A three-digit number indicating the number of characters to be output per printline, when the variable set is output using the EXTRACT mode of output processing. This number may be from 001 to 132.* If it exceeds 052 an advisory message (which should be ignored) results.
Col 17 = Comma.
Cols 18-XX = The Label to be used with the variable set data when it is output by the EXTRACT mode of output processing. The label is those characters between @ symbols. It may be from zero to 56 characters in length.

* Note: You may not obtain the full number of characters specified, as the program will only print to the last complete word within each line. Example - 100 characters requested per line - print position 98 holds the last character of a word. The next variable data to be printed is "know." Instead of the "n" being placed as the last character of the current line and the "ow" at the beginning of the next line, "now" is placed into print positions 1, 2, and 3 of the new line.

Figure 3-8. VSET Card.

"INDEX" Card*

* NOTE - Cross Indexing is described generally in Section II, beginning on page 2-19.

CROSS INDEX NAME (2-6)	CARD TYPE (8-12)	ELEMENT (GROUP/FIELD) (14-18)	,	SET ID (20)	,	REL. H.O.P. (22-25)	,	ELEMENT SIZE (27-30)	,	ARG LNG (32-33)	,	FUNCT LNGT (35-36)	,	SUBROUTINE NAME (38-43)	(44-72)
CMFLX	INDEX	DESTB	,	X	,	0034	,	0012	,	12	,	30	,	RTCIS	
CMFLY	INDEX	FLDID	,	n	,	ZZZZ	,	WWWW	,	nn	,	SS	,	FUBRS	

Col 01 = Blank

Cols 02-06 = The Name of the cross index is the first four characters of the file name, followed by X for first cross index of the file; Y if the second cross index of the file. A file may be cross indexed by a maximum of two elements, therefore there may be a maximum of two INDEX cards per file.

Col 07 = Blank.

Cols 08-12 = INDEX.

Col 13 = Blank.

Cols 14-18 = Name of the field or group (Element), by which to cross index the file. This element cannot be a component of the record ID for the file.

Col 19 = Comma.

Col 20 = Set ID of the element specified in columns 14 - 18. X = Fixed Set, 1-8 = Periodic Set No.

Col 21 = Comma.

Cols 22-25 = Relative position of high order character of the element specified in columns 14-18. (Relative to the first character of the four position record character count, if a fixed element. Relative to first character of PSSQ#, if a periodic element. In either case, count the first character as 0000.)

Figure 3-9. INDEX Card

Col 26 = Comma.

Cols 27-30 = Size of the element specified in columns 14-18 (as it appears in the file record).

Col 31 = Comma.

Cols 32-33 = The element by which the file is to be cross indexed is specified in cols 14-18. The size of that element (as it appears in the file record) is specified in cols 27-30. Argument is another term used to specify the element by which the file is cross indexed. The size of the argument (element) as it appears in the cross index is placed in cols 32-33. An FxxxS subroutine, (specified in columns 38-42), may be used to convert the data in the element from its file form into the form in which it will appear in the cross index, thus possibly changing its size. If a FxxxS subroutine is used, the output size of the argument (element) from the subroutine is placed in cols 32-33. If no FxxxS subroutine is used, the data in the argument (element) will appear in the cross index just as it appears in the file, and cols 32 and 33 will contain the size previously placed in cols 29 and 30 of this card.

Col 34 = Comma.

Cols 35-36 = The cross index is a list of record ID's, by unique data content of the element by which the file is cross indexed. Each of these record ID's is called a function of the cross index. The size of the function (record ID size) is placed in cols. 35-36.

Col 37 = Comma.

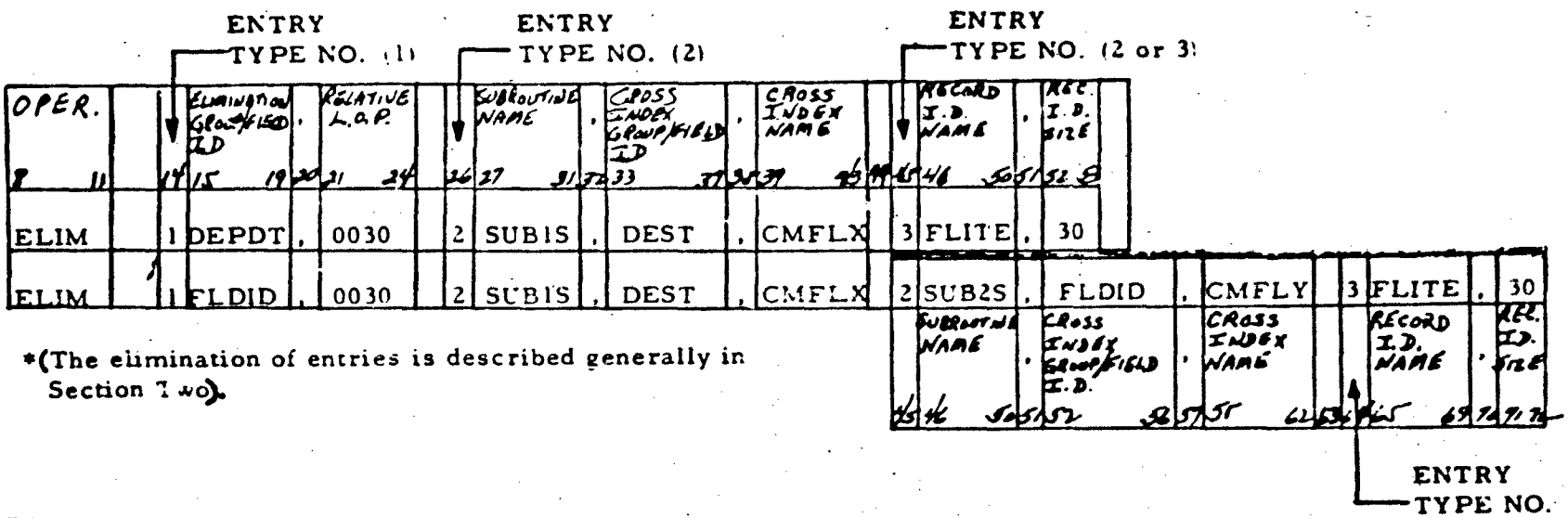
Cols 38-42 = If the data in the element (argument) by which the file is cross indexed is to be converted before placement in the cross index, insert the name of the conversion subroutine (first letter "F", last letter "S") FxxxS in cols 38-42.

Col 43 = Blank.

Cols 44-72 = Any comments desired, or blanks.

Figure 3-9. (Continued), INDEX Card

"ELIM" Card*

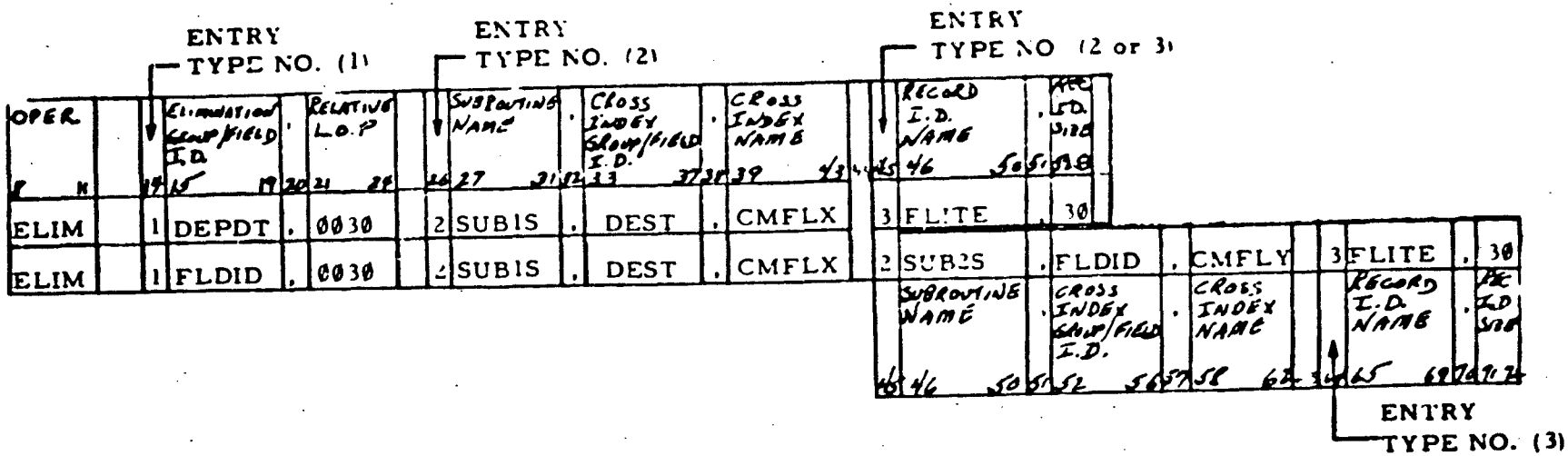


*(The elimination of entries is described generally in Section Two).

Singly Indexed File	Doubly Indexed File
Col 45 = "3" for entry type three. (Record ID entry.) Cols 46-50 = Record ID name Col 51 = Comma Cols 52-53 = Size of record ID Col 54 = Blank Cols 55-72 = Blanks, or any comments desired.	Col 45 = "2" for second type two entry. (Second cross-index entry.) Cols 46-50 = Subroutine to convert field name for cross-index processor. Col 51 = Comma Cols 52-56 = The "other" Group/Field ID by which the file is indexed. (The same ID specified in columns 14-18 of the "other" INDEX card.) Col 57 = Comma Cols 58-62 = Name of the "other" cross index. Col 63 = Blank Col 64 = "3" for entry type three. (Record ID entry.) Cols 65-69 = Record ID name. Col 70 = Comma Cols 71-72 = Size of record ID.

Figure 3-10. ELIM Card.

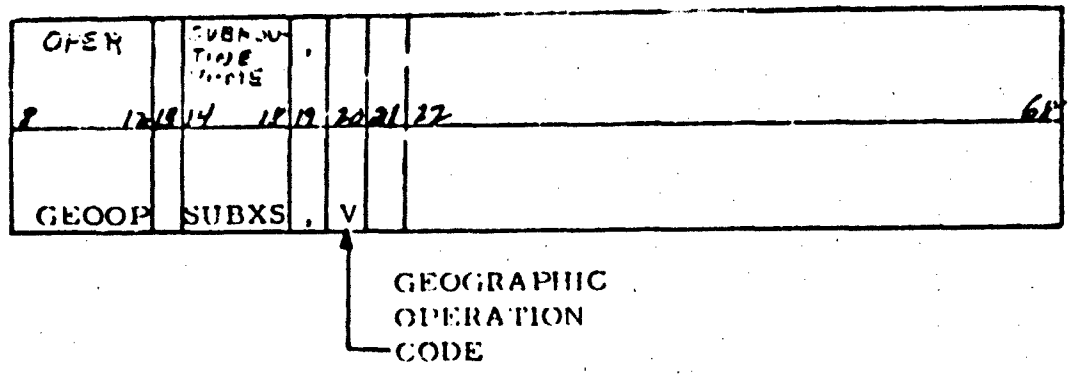
"ELIM" Card (The elimination of entries is described generally in Section Two.)



Cols 01-07 = Blanks.
Cols 08-11 = ELIM.
Cols 12-13 = Blanks.
Col 14 = "1" for entry type one. (Elimination field entry.)
Cols 15-19 = Group/Field ID by which the cross index is to have entries selectively eliminated. This must be a group/field contained within the record ID, and must not be over 15 characters in length.
Col 20 = Comma.
Cols 21-24 = "Relative Low Order Position." Relative position of the low order character of the group/field specified in columns 15-19. (Relative to the first character of the first fixed field after the record character count, which is counted as 0001.)
Col 25 = Blank.
Col 26 = "2" for entry type two. (Cross-index entry.)
Cols 27-31 = Subroutine to convert field name for cross-index processor.
Col 32 = Comma.
Cols 33-37 = Group/Field ID by which the file is indexed. (The same ID specified in columns 14-18 of the INDEX card.)
Col 38 = Comma.
Cols 39-43 = Name of the cross index (The name in columns 2-6 of the INDEX card).
Col 44 = Blank.

Figure 3-10. ELIM Card.

"GEOOP" Card (The geographic operation code information, following the elimination card, is also described in Section Two.)



Cols 01-07 = Blanks.
Cols 08-12 = GEOOP.
Col 13 = Blank.
Cols 14-18 = Subroutine name (must end in S) corresponding to operation code in column 20. One subroutine may be related to more than one operation code. Used in retrieval to process absolute values against record content (field name).
Col 19 = Comma
Col 20 = Geographic operation code of 1 character related to subroutine name in columns 14-18. May not refer to more than one subroutine.
Col 21 = Blank, used to separate this combination from next subroutine and operation code.
Cols 22-68 = Up to 6 additions (total 7) subroutine and operation code combinations are allowed. Each combination is separated from the next by a blank.
NOTE: As the geographic operation code may refer to only one subroutine and as Mark II only has two codes (V and E) we will never have more than two entries on the GEOOP Card. The other five possible entries on the card are for future use of the system.
Cols 69-72 = Blank.

Figure 3-11. GEOOP Card.

h. Allowable Card Sequence.

(1) In general, the FSJOB card will be the first card of the input deck to FS.* Following the FSJOB card will be the SUB, TAB and EDIT cards in any order. Next come the FIELD cards which define the fields comprising the record ID, followed by a GROUP card, grouping these fields. Then come additional FIELD and GROUP cards, in the order of their desired placement in the file. (Note that a GROUP card must immediately follow the FIELD cards which define the fields to be grouped.) After all fields and groups have been specified, a VSET card may appear if the file is to have provision for a variable set. Next in order is the INDEX card, if the file is to be cross indexed. A maximum of two INDEX cards may be used for one file. An ELIM card must follow the INDEX card(s). The GEOOP card is next if you use the special geographic operator.

(2) At this point, there may either appear another FSJOB card or an ENDFS card. A new FSJOB card signifies the end of the input cards for the structuring of one file's FFT, and the beginning of the input cards for the next file's FFT. An ENDFS card signifies the end of all input cards to FS.

(3) The next two figures specify precisely the allowable sequence of FS input cards.

* The exception is the BYPASS card, in which case no card except the BYPASS card may appear in the FS input deck. After the BYPASS card should appear the input cards for file revision.

Reading DOWN in the table gives the cards which may immediately Follow the card at the top ....e.g., a VSET card may be immediately followed by an FSJOB, INDEX, GEOOP, or ENDFS card.

FSJOB Card	BYPASS Card	SUB TAB EDIT Card	FIELD C Card	GROUP Card	FIELD X(1,2,---,8) Card	VSET Card	INDEX Card	ELIM Card	GEOOP	ENDFS Card
				GEOOP	GEOOP					
SUB TAB EDIT		SUB TAB EDIT	FSJOB	FSJOB	FSJOB	FSJOB		GEOOP	FSJOB	
			FIELD C	FIELD C	GROUP	INDEX	INDEX	FSJOB	ENDFS	
FIELD C		FIELD C	GROUP	GROUP	(Only if in seq)	GEOOP	ELIM	ENDFS		
			FIELD X (1,2-8)	FIELD X (1,2-8)	FIELD X (1,2-8)	ENDFS				
			VSET	VSET	VSET					
			GEOOP	INDEX	INDEX					
			ENDFS	ENDFS	ENDFS					

Figure 3-11A. "Cards Which May Immediately Follow" Table.

Reading DOWN in the table gives the cards which may immediately precede the card at the top ....e.g., FSJOB may be immediately preceded by any FIELD, GROUP, VSET, INDEX, or ELIM, GEOOP Card.

FSJOB Card	BYPASS Card	SUB TAB EDIT Card	FIELD C Card	GROUP Card	FIELD X (1,2, ---,8) Card	VSET Card	INDEX Card	ELIM	GEOOP	ENDFS
FIELD C GROUP FIELD X (1,2-8) VSET ELIM GEOOP		FSJOB SUB TAB EDIT	FSJOB SUB TAB EDIT FIELD C GROUP	FIELD C GROUP FIELD X (1,2-8)	FIELD C GROUP (ONLY IF IN SEQ) FIELD X (1,2-8)	FIELD C GROUP FIELD X (1,2-8)	GROUP FIELD X (1,2-8) VSET INDEX	INDEX	FIELD C GROUP FIELD X (1,2-8) VSET ELIM	FIELD C GROUP FIELD X (1,2-8) VSET ELIM GEOOP

Figure 3-11B. "Cards Which May Immediately Precede" Table.

i. The CMFLA File Structuring Run Example. Figures 3-13 through 3-22 comprise ten pages which illustrate the file structuring run which created the sample file CMFLA. The use of every type of input card (except BYPASS) is illustrated. The first five pages are the input card listing. The next five pages are the listing of the file format table created by file structuring. A careful analysis of this sample file structuring run is strongly recommended. While analyzing the sample run, reference should be made back to the FS input card specifications and also to the following information which gives an explanation of the makeup of the file format table.

j. The File Format Table.

(1) The data content of each file is organized, manipulated and processed in accordance with the file specifications contained in the file format table (FFT) for that file. The FFT contains not only the specifications as to the internal format of the file, but also contains specifications relative to the external formats and conversions between the external and internal formats. The FFT may be thought of as the link between the user's point of view of the file (as an ordered collection of fields, groups, and sets with associated names, conversions, labels, and other specifications) and the highly detailed programming oriented specifications required by 1410 Formatted File System program components.

(2) The FFT is made up of nine logical records, as illustrated by figure 3-12.

Logical Record 1 - "ID Record" Contains the five character mnemonic of the data file.

Logical Record 2 - "Detail Information Table." Type entry ID made up of coded field/group/set/subset entries as follows: (1 char)

1 = Record Control Field/Group

2 = Fixed (Non-Control) Field/Group.

3 = Periodic Field/Group.

4 = Variable Set.

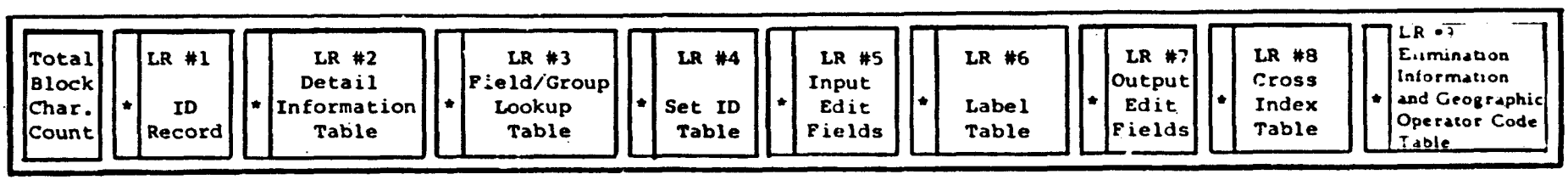
5 = Periodic Subset.

6 = Periodic Set Control Field (PSCTn).

7 = Variable Set Control Field (VSCTL).

Entry number. Specifies the sequence of entries to the LR #2 of the FFT. (2 char)

The Logical Records of the File Format Table appear in this order in one physical block on tape, disk, or in core after read-in.



*The first four characters of each logical Record are the character count for that Logical Record.

Figure 3-12. FFT Logical Record Layout.

Relative high order position (HOP) in the data record (4 char).

For fixed fields/groups, this entry refers to the position in the record relative to the first character in the record (which is counted as relative zero).

For periodic fields/groups and for the variable set this entry refers to position in the data record relative to the first character of the subset or variable set in which they are included. (The first character of a subset or the variable set will always be relative zero.)

Number of characters in the field/group/subset (4 char). (For the variable set this indicates the number of characters to be printed per line by EXTRACT mode.)

Set ID of the field/group/subset (1 char).

X = Fixed Set.

1 = Periodic Set 1.

2-8 = Periodic Set 2-8.

9 = Variable Set.

Input type parameters (2 char minimum variable).

The first character indicates the source of the entry input (1-9).

1 = Retrieval.

2 = File Maintenance Internal Format.

3-9 = External Format.

$ = All inputs to this entry are to be processed identically.

The second character indicates:

1 = Alphameric information is contained in the field/group/subset.

2 = Numeric information is contained in the field/group/subset.

3 = Input information must be edited.

The first alphabetic character of a five-character name of the subroutine that is to be used to process the inputs to this field.

The third character contains:

The next entry to specify data type (alphameric or numeric) for a second of "n" inputs to the field/group/subset (i.e., a second input type parameter entry).

If position number two contains a 3, this position will contain the first digit of a three-digit low order relative address of an edit field contained in LR #5 (the input edit record).

If a subroutine is required to process input data to a field/group/subset, this position will contain the second character of the pertinent subroutine name begun in position number two.

Logical Record 3 - Field/Group Lockup Table.

LR #3 is entered with a defined field/group mnemonic.

The table gives the addresses of this mnemonic in LR #2 and LR #6.

Logical Record 4 - Set ID Table.

The contents are:

Retrieval Logic Codes.

I = Inter Mode.

S = Scan Mode.)
) Forms of
N = Non-Scan Mode.) INTRA Mode
)
T = Terminate Mode.)

The relative high order position of the pertinent set control field in the data record.

The size of the fixed set or periodic subset.

Set ID.

X = Fixed

1 = Periodic Set 1.

2-8 = Periodic Set 2-8.

9 = Variable Set.

Logical Record 5 - The Input Edit Record.

The actual input edit control word.

A three-character field specifying the maximum field size that can be processed using this edit control word.

Logical Record 6 - Label Table.

The output size of the field/group.

The size of the label relative to the field/group size (+ or -).

The operation ID.

Blank = No operation to be performed on the field/group prior to output.

1 = Field/group is to be processed by a subroutine prior to output.

2 = Field/group is to be edited prior to output.

The operator address, if applicable, and the actual output label.

For subroutine processing, the label is preceded by the subroutine name.

For editing, the output label is preceded by the relative low order position of the edit word in LR #7.

For output with no prior processing, this area contains the output label only.

Logical Record 7 - The Output Edit Record.

The actual output edit word.

A three-character field specifying the maximum field size that can be processed using this edit word.

Logical Record 8 - The Cross-Index Table.

FxxxS or RxxxS subroutine, five-character fields.

Set ID.

X = Fixed Set.

1 - Periodic Set 1.

2-8 = Periodic Set 2-8.

A four-character field specifying the size of the element. Used (as an argument) to cross index the file as it appears in the file record.

A four-character field giving the high order position of the element used as the argument.

A five-character field specifying the name of the element used as the argument to cross index the file as it appears in the FFT.

A two-character field specifying the size of the argument as it appears in the cross-index table.

A two-character field specifying the size of the function (record ID size).

A five-character field specifying the first four characters of the file name, followed by an X or Y.

Logical Record 9, Part 1 - Elimination Information Table.

Table Entry Type 1:

A four-character field specifying the relative low order position of the element by which entries in the cross index may be selectively eliminated from consideration for retrieval purposes.

A five-character field specifying the name of the element (as it appears in the FFT) by which entries may be selectively eliminated from the cross-index table for retrieval purposes.

Table Entry Type 2:

A five-character field specifying the name of the cross index.

A five-character field specifying the name of the element by which the file is cross indexed.

A five-character field specifying the name of the subroutine to convert field names for cross-index processor.

NOTE: There will be two "Table Entry Type 2" entries in LR #9 if the file is doubly indexed.

Table Entry Type 3:

A two-character field specifying the length of the record ID.

A five-character field specifying the name of the record ID (record control group).

Logical Record 9, Part 2 - Geographic Operation Code Informati

A five-character field specifying the subroutine name used in retrieval to process absolute values against record content (field name).

A five-character field of which the first four characters are nines, filled in by the FFS program, the fifth character is the geographic operations code.

Six other possible entries can be repeated as above.

```
FILE STRUCTURING                    *** CREATE CMFLA ***        COPY 01 OF 02        DATE 17NOV     PAGE 01

                                                                                                  001CMFLA
      FSJOB CREATE.CMFLA.DATE.NOOK.LST1                                                  ADVISORY MESSAGE
                  FFT ALREADY IN LIBRARY. IT WILL BE CHANGED IN LIB. UPDATE RUN.

          * FILE NAME   COMMERCIAL FLIGHTS FILE                                          002CMFLA
          * FILE ID     CMFLA                                                          003CMFLA
          * OBJECTIVE   THE CMFLA FILE IS A SAMPLE FILE USED THROUGHOUT THE            004CMFLA
          *             FILE GENERATION AND MAINTENANCE MANUAL, AS WELL AS             005CMFLA
          *             THE RETRIEVAL AND OUTPUTS MANUAL.  ITS USE PROVIDES            006CMFLA
          *             CONTINUITY BETWEEN SUBJECTS AND HELPS TO ILLUSTRATE            007CMFLA
          *             SOME OF THE CONCEPTS AND SPECIFICATIONS OF THE IDMS            008CMFLA
          *             1410 FES MARK 5. CMFLA IS PRIMARILY A TRAINING AID.            009CMFLA
          ************************************************************************** 010CMFLA
          * THE PURPOSE OF THIS RUN IS TO ILLUSTRATE THE USE OF FILE STRUCTURING * 011CMFLA
          * IN IMPLEMENTING A DATA FILE, AND THE MAKEUP OF A FILE FORMAT TABLE.  * 012CMFLA
          ************************************** ************************************* 013CMFLA
          *                                                                        014CMFLA
          * THE FOLLOWING IS USED FOR INPUT EDITING                                015CMFLA

                                                                                           016CMFLA
FLTNO EDIT  * 0 *  SUPPRESSES UP TO 2 LEADING ZEROS.  USED FOR FLTNO.

          * THE FOLLOWING ARE USED FOR OUTPUT EDITING                              017CMFLA

                                                                                           018CMFLA
PNLTY EDIT  '$   .'                                                                        019CMFLA

OEDCO EDIT  '  +DEG+  +MIN+  +SEC+ .+   +DEG+  +MIN+  +SEC+ .'

          *                          USED FOR OCOOR AND TCOOR.                   020CMFLA

                                                                                           021CMFLA
GEREF EDIT  '  D  M  S+ .   D  M  S+ .  D  M  S+ .   D  M  S+ .'

          *                                                  FOR GEREF.          022CMFLA

                                                                                           023CMFLA
LXOFY EDIT  'LEG+ +OF+ '                 USED FOR LXOFY

          ************************************************************************** 024CMFLA
          * THE FOLLOWING IS USED FOR THE INPUT CONVERSION OF THE FLTYP FIELD.   * 025CMFLA
          * IT CONVERTS THE NUMBERS 1 THRU 5 TO P, C, D, T, OR X RESPECTIVELY.   * 026CMFLA

                                                                                           027CMFLA
IFLTS TAB   0061.001                                                       ADVISORY MESSAGE
                  NOT IN LIBRARY.

          *THE FOLLOWING OUTPUT CONVERSION SUBROUTINE PLACES A COMMA JUST PRIOR TO  028CMFLA
          *THE LOW ORDER DIGIT OF THE DATA AND THEN ADDS TWO TRAILING ZEROS TO IT.  029CMFLA

                                                                                           030CMFLA
HUNDS SUB   0061.007      USED FOR ALTID, FUELL, AND GRUNT                         ADVISORY MESSAGE
                  NOT IN LIBRARY.
```

Figure 3-13

```
          * THE FOLLOWING SUBROUTINE IS USED FOR OUTPUTTING EITHER THE FLTYP FIELD      031CMFLA
          * OR THE DEPDT GROUP.  IT CONVERTS THE DATA IN FLTYP FROM EITHER P TO         032CMFLA
          * PASSG, C TO CARGO, D TO DEDHD, T TO CHART, OR X TO OTHER.  IF THE DATA      033CMFLA
          * FROM DEPDT IS BEING CONVERTED, IT OUTPUTS THE SCHED DEPT. DATE/TIME IN     034CMFLA
          * THE FORM OF  DD MMM YYYY HHMM,  E.G. 19 NOV 1965 1723.                     035CMFLA

OPTCS SUB   0259.005=1=,016=2=                                                                  036CMFLA
                    NOT IN LIBRARY.                                                  ADVISORY MESSAGE

          ********************************************************************************    037CMFLA
          * THE FOLLOWING COMPRISE THE FIXED SET.                                             038CMFLA

ANAME FIELD 008,C, ,*$,ALPHA,*      ,'AIRLINE'                                                  039CMFLA

ORGIN FIELD 012,C, ,*$,ALPHA,*      ,'FLT ORIGIN'                                               040CMFLA

DYEAR FIELD 002,C, ,*$,NUMER,*      ,'YEAR'                                                     041CMFLA

DMNTH FIELD 002,C  ,*$,NUMER,*      ,'MONTH'                                                    042CMFLA

DDATE FIELD 002,C, ,*$,NUMER,*      ,'DAY'                                                      043CMFLA

DHOUR FIELD 004,C, ,*$,NUMER,*      ,'HOUR'                                                     044CMFLA

DEPDT GROUP DYEAR,DMNTH,DDATE,DHOUR,*$,NUMER,*OPTCS=2=,'DEPARTURE DATE TIME'                    046CMFLA
                                                                                        SEQUENCE ERROR.
FLITE GROUP ANAME,ORGIN,DEPDT,*$,ALPHA,*     ,'FLIGHT'

CREDT FIELD 006,X, ,*$,NUMER,*      ,'CREATE DATE'

CHGDT FIELD 006,X, ,*$,NUMER,*      ,'CHANGE DATE'

DEST  FIELD 012,X, ,*$,ALPHA,*      ,'DESTINATION'                                              048CMFLA

FLTNO FIELD 003,X, ,*$,FLTNO,*      ,'FLT'                                                      049CMFLA

ACTYP FIELD 004,X, ,*$,ALPHA,*      ,'A/C TYPE'                                                 050CMFLA

FLTYP FIELD 001,X, ,*1,ALPHA,2,ALPHA,3,IFLTS,*OPTCS=1=,'FLIGHT TYPE'                            051CMFLA

CAPCY FIELD 003,X, ,*$,NUMER,*      ,'CAPACITY'                                                 052CMFLA

          *                                                                                 053CMFLA
          *                                                                                 054CMFLA
          ********** THE FOLLOWING COMPRISE PERIODIC SET ONE **********                      055CMFLA
```


Figure 3-14

```
FILE  STRUCTURING                              ***  CREATE  CMFLA  ***        COPY 01 OF 02        DATE  17NOV     PAGE  03

LORIG FIELD 012.1.S.*$.ALPHA.*      .*LEG ORIGIN*                                                                  056CMFLA
LTERM FIELD 012.1.S.*$.ALPHA.*      .*LEG TERMINATION*                                                             057CMFLA
OCOOR FIELD 015.1.S.*$.ALPHA.*DEDCO.*LEG ORIGIN COORD*                                                             058CMFLA
TCOOR FIELD 015.1.S.*$.ALPHA.*DFDCO.*LEG TERM. COORD*                                                              059CMFLA
GEREF GROUP OCOOR.TCOOR.*$.ALPHA.*GEREF.*LEG ORIGIN COORDINATES*                                                   060CMFLA
LXOFY FIELD 002.1.S.*$.NUMER.*LXOFY.*LEG X OF Y*                                                                   061CMFLA
TDATE FIELD 002.1.S.*$.NUMER.*      .*TO DATE*                                                                     062CMFLA
TTIME FIELD 004.1.S.*$.NUMER.*      .*TO TIME*                                                                     063CMFLA
TOD/T GROUP TDATE.TTIME.*$.NUMER.*      .*TO DT/TIME*                                                              064CMFLA
LDATE FIELD 002.1.S.*$.NUMER.*      .*LOG DATE*                                                                    065CMFLA
LTIME FIELD 004.1.S.*$.NUMER.*      .*LOG TIME*                                                                    066MFLA
LD/T  GROUP LDATE.LTIME.*$.NUMER.*      .*LOG DT/TIME*                                                             067CMFLA
ALTTD FIELD 003.1.S.*$.NUMER.*HUNDS.*AVG ALT*                                                                      068CMFLA
ONBRD FIELD 003.1.S.*$.NUMER.*      .*ON BOARD*                                                                    069CMFLA
FUELL FIELD 003.1.S.*$.NUMER.*HUNDS.*FUEL LOADED*                                                                  070CMFLA
GROWT FIELD 004.1.S.*$.NUMER.*HUNDS.*TAKEOFF WEUGHT*                                                               071CMFLA
        *** THE FOLLOWING CARD CAUSES A NO. 5 TYPE ID ENTRY IN LR2 OF THE FFT **                    072CMFLA
PSET1 GROUP PSS01.LORIG.LTERM.GEREF.LXOFY.TOD/T.LD/T .ALTTD.ONBRD.FUELL.GROWT.*$.ALPHA.*     .*PERIODIC SET ONE*          074CMFLA
                                                                                                          SEQUENCE ERROR.
                    EXCEEDS FILE MAINTENANCE EXTERNAL FORMAT CARD INPUT LIMITATION.                       ADVISORY MESSAGE
                    TOO LARGE TO BE QUERIED EXCEPT WITH OVP OPERATOR.                                     ADVISORY MESSAGE
                    EXCEEDS SDP LIMIT FOR MOVING TO LITERAL.                                              ADVISORY MESSAGE

        *                                                                                          075CMFLA
        *                                                                                          076CMFLA
        ********************* THE FOLLOWING COMPRISE PERIODIC SET TWO*********************          077CMFLA

LOCAN FIELD 012.2.S.*$.ALPHA.*      .*LOCATION*                                                                    078CMFLA
```

Figure 3-15

```
..OMM FIELD 001,2,S,*$,NUMER,*      ,'NAV/COMM'                                                    079CMFL1
FRAME FIELD 001,2,S,*$,NUMER,*      ,'AIRFRAME'                                                    080CMFLA
ELSYS FIELD 001,2,S,*$,NUMER,*      ,'ELEC SYS'                                                    081CMFLA
HYSYS FIELD 001,2,S,*$,NUMER,*      ,'HYDR SYS'                                                    082CMFLA
ENGIN FIELD 001,2,S,*$,NUMER,*      ,'ENGINE'                                                      083CMFLA
LGEAR FIELD 001,2,S,*$,NUMER,*      ,'LDG GEAR'                                                    084CMFLA
ENMNT FIELD 001,2,S,*$,NUMER,*      ,'ENV CNTL'                                                    085CMFLA
RPLCD FIELD 001,2,S,*$,NUMER,*      ,'A/C REPL'                                                    086CMFLA
OTHER FIELD 001,2,S,*$,..UMER,*     ,'OTHER'                                                       087CMFLA
MAINT GROUP NCOMM,FRAME,ELSYS,HYSYS,ENGIN,LGEAR,ENMNT,RPLCD,OTHER,*$,NUMER,*     ,'UNSCH MAINT'   089CMFLA
                                                                                  SEQUENCE ERROR.

MEALS FIELD 003,2,S,*$,NUMER,*      ,'MEALS PURCH'                                                 090CMFLA
ROOMS FIELD 003,2,S,*$,NUMER,*      ,'PASS BILLET'                                                 091CMFLA
PNLTY FIELD 003,2,S,*$,NUMER,*PNLTY,'FRGHT PNLTY'                                                  092CMFLA
RLOSS GROUP MEALS,ROOMS,PNLTY,*$,NUMER,*     ,'REVENUE LOSS'                                      093CMFLA

        *** THE FOLLOWING CARD CAUSES A NO. 5 TYPE ID ENTRY IN LR2 OF THE FFT **          094CMFLA

PSET2 GROUP PSSQ2,LOCAN,MAINT,RLOSS,*$,ALPHA,*     ,'SET 2'                                        095CMFLA

        *                                                                                  096CMFLA
        *                                                                                  097CMFLA
        ** ********** FOLLOWING IS THE VARIABLE SET SPECIFICATION**************            098CMFLA

REMRK VSET  120,'R E M A R K S   R E M A R K S   R E M A R K S'                            099CMFLA
                EXCEEDS FILE MAINTENANCE EXTERNAL FORMAT CARD INPUT LIMITATION.           ADVISORY MESSAGE
                TOO LARGE TO BE QUERIED EXCEPT WITH OVP OPERATOR.                         ADVISORY MESSAGE
                EXCEEDS SOP LIMIT FOR MOVING TO LITERAL.                                  ADVISORY MESSAGE

        *                                                                                  100CMFLA
        *                                                                                  101CMFLA
        * THE FOLLOWING CARD SPECIFIES THAT A CROSS INDEX TABLE OF ARGUMENTS AND           102CMFLA
        * FUNCTIONS IS TO BE BUILT AND MAINTAINED FOR THIS FILE. FOR EACH UNIQUE          103CMFLA
        * DATA CONTENT OF THE DEST  FIELD THERE IS AN ARGUMENT. THE FUNCTIONS OF         104CMFLA
        * THE ARGUMENTS ARE THE RECORD IDS OF THE RECORDS HAVING THE ARGUMENT IN         105CMFLA
        * THEIR DEST  FIELD.                                                            106CMFLA
```


Figure 3-16

```
                                                                                    107CMFLA
CMFLX INDEX DEST .X.U046.0012.12.30.FXXXS                                           ADVISORY MESSAGE
                  NOT IN LIBRARY.

                                                                                    108CMFLA
      *                                                                             109CMFLA
      * THE FOLLOWING SPECIFICATION ALLOWS THE SELECTIVE ELIMINATION OF RECORD      110CMFLA
      * IDS *FUNCTIONS* IN THE DEST  CROSS INDEX TABLE FROM CONSIDERATION *FOR      111CMFLA
      * RETRIEVAL PURPOSES*. BASED UPON THE DATA CONTENT OF THE DEPOT GROUP.
                                                                                    112CMFLA
      ELIM  1DEPOT.0030 2SUBIS.DEST .CMFLX 3FLITE.30
                                                                                    113CMFLA
      GROUP SUBXS.V SUBXS.E

                                                                                    114CMFLA
      *
```


Figure 3-17

LOGICAL RECORD 1 FFT ID RECORD CHARACTER COUNT 0015 FILE NAME CMFLA

LOGICAL RECORD 2 DETAIL INFORMATION TABLE CHARACTER COUNT 0841

ENTRY M.O.P.	TYPE ID.	ENTRY NUMB.	REL M.O. POS IN DATA REC.	NO. CHAR. IN FLD-GRP OR SUBSET	SET ID.	INPUT TYPE PARAMETERS
00004	7	+1	0000	0004	X	$2)
00019	1	+2	0004	0008	X	$1)
00034	1	+3	0012	0012	X	$1)
00049	1	+4	0024	0002	X	$2)
00064	1	+5	0026	0002	X	$2)
00079	1	+6	0028	0002	X	$2)
00094	1	+7	0030	0004	X	$2)
00109	1	+8	0024	0010	X	$2)
00124	1	+9	0004	0030	X	$1)
00139	2	AC	0034	0006	X	$2)
00154	2	A1	0040	0006	X	$2)
00169	2	A2	0046	0012	X	$1)
00184	2	A3	0058	0003	X	$3009)
00202	2	A4	0061	0004	X	$1)
00217	2	A5	0065	0001	X	11213(FLTS)
00240	2	A6	0066	0003	X	$2)
00255	6	A7	0069	0008	X	12)
00270	3	A8	0000	0003	1	$2)
00285	3	A9	0003	0012	1	$1)
00300	3	B0	0015	0012	1	$1)
00315	3	B1	0027	0015	1	$1)
00330	3	B2	0042	0015	1	$1)
00345	3	B3	0027	0030	1	$1)
00360	3	B4	0057	0002	1	$2)
00375	3	B5	0059	0002	1	$2)
00390	3	B6	0061	0004	1	$2)
00405	3	B7	0059	0006	1	$2)
00420	3	B8	0065	0002	1	$2)
00435	3	B9	0067	0004	1	$2)
00450	3	C0	0065	0006	1	$2)
00465	3	C1	0071	0003	1	$2)
00480	3	C2	0074	0003	1	$2)
00495	3	C3	0077	0003	1	$2)
00510	3	C4	0080	0004	1	$2)
00525	5	C5	0000	0084	1	$1)
00540	6	C6	0077	0008	X	12)
00555	3	C7	0000	0003	2	$2)
00570	3	C8	0003	0012	2	$1)
00585	3	C9	0015	0001	2	$2)
00600	3	D0	0016	0001	2	$2)
00615	3	D1	0017	0001	2	$2)
00630	3	D2	0018	0001	2	$2)
00645	3	D3	0019	0001	2	$2)

Figure 3-18

00660	3	D4	0020	0001	2	$2)
00675	3	D5	0021	0001	2	$2)
00690	3	D6	0022	0001	2	$2)
00705	3	D7	0023	0001	2	$2)
00720	3	D8	0015	0009	2	$2)
00735	3	D9	0024	0003	2	$2)
0075C	3	E0	0027	0003	2	$2)
00765	3	E1	0030	0003	2	$2)
00780	3	E2	0024	0009	2	$2)
00795	5	E3	0000	0033	2	$1)
00810	7	F4	0085	0008	X	12)
00825	4	E5	0000	G120	9	$1)

LOGICAL RECORD 3 FIELD/GROUP LOOKUP TABLE CHARACTER COUNT 0721

ENTRY N.O.P.	REL ADDR IN LR 4	REL ADDR IN LR 2	FIELD OR GROUP ID
00005	0004	0004	RECCT
00018	0023	0019	ANAME
00031	0039	0034	ORGIN
00044	0058	0049	DYEAR
00057	0071	0064	DMNTH
00070	0085	0079	DDATE
00083	0097	0094	DHOUR
00096	0110	0109	DEPDT
00109	0143	0124	FLITE
00122	0158	0139	CREDT
00135	0178	0154	CHGDT
00148	019A	0169	DEST
00161	021A	0184	FLTNO
00174	0230	0202	ACTYP
00187	0247	0217	FLTYP
00200	0272	0240	CAPCY
00213	0299	0255	PSCTI
00226	0306	0270	PSSQI
00239	0321	0285	LORIG
00252	0340	0300	LTERM
00265	0364	0315	OCOOR
00278	0392	0330	TCOOR
00291	0419	0345	GERFF
00304	0453	0360	LXOFY
00317	0475	0375	TDATF
00330	0491	0390	TTIMF
00343	0507	0405	TOD/T
00356	0526	0420	LDATE
00369	0543	0435	LTIME
00382	0560	0450	LD/T
00395	0580	0465	ALTTD
00408	0601	0480	ONBRD

Figure 3-19

00421	0618	0495	FUFLL
00434	0643	0510	GROWT
00447	0671	0525	PSET1
00460	0696	0540	PSCT2
00473	0713	0555	PSSQ2
00486	0728	0570	LOCAN
00499	0745	0585	NCOMM
00512	0762	0600	FRAME
00525	0779	0615	ELSYS
00538	0796	0630	HYSYS
00551	0813	0645	LOGIN
00564	0828	0660	LGEAR
00577	0845	0675	ENMNT
00590	0862	0690	RPLCD
00603	0879	0705	OTHER
00616	0893	0720	MAINT
00629	0913	0735	MEALS
00642	0933	0750	ROOMS
00655	0953	0765	PNLTY
00668	0976	0780	RLOSS
00681	0997	0795	PSET2
00694	1011	0810	VSCTL
00707	1028	0825	REMRK

LOGICAL RECORD 4 SET ID TABLE CHARACTER COUNT 0050

ENTRY M.O.P.	LOGIC ID.	SET CTL FLD NOP	FIXED SET OR SUBSET SIZE	SET ID - ONE CHAR AND BLANK
00005		DATE	0093	X
00016	S	0069	0084	1
00027	S	0077	0033	2
00038		0085		9

LOGICAL RECORD 5 INPUT EDIT FIELDS CHARACTER COUNT 0011

ENTRY M.O.P. THE LAST THREE CHARACTERS OF EACH ENTRY GIVE THE MAXIMUM SIZE OF A FIELD WHICH CAN BE EDITTED BY THAT ENTRY.

00004 0 003

Figure 3-20

LOGICAL RECORD 6 LABEL TABLE CHARACTER COUNT 1085

ENTRY W.O.P.	OUTPUT SIZE	RELATIVE LABEL SIZE	OP ID	OPERATOR ADDRESS, IF APPLICABLE, AND ACTUAL LABEL
00004	0004	00F		CHAR COUNT)
00023	0008	00J		AIRLINE)
00039	0012	00K		FLT ORIGIN)
00058	0002	008		YEAR)
00071	0002	00C		MONTH)
00085	0002	00A		DAY)
00097	0014	00+		HOUR)
00110	0016	00C	1	OPTCSDEPARTURE DATE TIME)
00143	0030	02M		FLIGHT)
00158	0006	00E		CREATE DATE)
00178	0006	00E		CHANGE DATE)
00198	0012	00J		DESTINATION)
00218	0003	00+		FLT)
00230	0004	00D		A/C TYPE)
00247	0005	00F	1	OPTCSFLIGHT TYPE)
00272	0003	00F		CAPACITY)
00249	0008	00+		PSCT 1)
00306	0003	00C		PSSQ 1)
00321	0012	00K		LEG ORIGIN)
00340	0012	00C		LEG TERMINATION)
00364	0048	03K	2	055LEG ORIGIN COORD)
00392	0048	03L	2	055LEG TERM. COORD)
00419	0050	020	2	108LEG ORIGIN COORDINATES)
00453	0010	00+	2	121LEG X OF Y)
00475	0002	00F		TO DATE)
00491	0004	00C		TO TIME)
00507	0006	00D		TO DT/TIME)
00526	0002	00F		LDG DATE)
00543	0004	00D		LDG TIME)
00560	0006	00F		LDG DT/TIME)
00580	0007	00+	1	HUNDSAVG ALT)
00601	0003	00F		ON BOARD)
00618	0007	00D	1	HUNDSFUEL LOADED)
00643	0007	00G	1	HUNDSTAKEOFF WEUGHT)
00671	0084	060		PERIODIC SET ONE)
00696	0008	00+		PSCT 2)
00713	0003	00C		PSSQ 2)
00728	0012	00M		LOCATION)
00745	0001	00G		NAV/COMM)
00762	0001	00G		AIRFRAME)
00779	0001	00G		ELEC SYS)
00796	0001	00G		HYDR SYS)
00813	0001	00E		ENGINE)
00828	0001	00G		LDG GEAR)
00845	0001	00G		ENV CNTL)
00862	0001	00G		A/C REPL)

Figure 3-21

```
00879     0001        00D          OTHER)
00893     0009        00B          UNSCH MAINT)
00913     0003        00H          MEALS PURCH)
00933     0003        00H          PASS BILLET)
00953     0005        00F      7   129FRGHT PNLTY)
00976     0009        00C          REVENUE LOSS)
00997     0033        020          SET 2)
01011     0008        00+           VSCTL  )
01028     0120        07L          R E M A R K S   R E M A R K S   R E M A R K S)
```

LOGICAL RECORD 7 OUTPUT EDIT FIELDS CHARACTER COUNT 0131

```
ENTRY      THE LAST THREE CHARACTERS OF EACH ENTRY GIVE THE MAXIMUM
H.O.P.    SIZE OF A FIELD WHICH CAN BE EDITTED BY THAT ENTRY.
 00005   +DEG+  +MIN+  +SEC+ .+   +DEG+  +MIN+  +SEC+ .015
 00056   D  M  S+ .   D  M  S+ .  D  M  S+ .   D  M  S+ .030
 00109 LEG+ +OF+ 002
 00122 $   .003
```

LOGICAL RECORD 8 CROSS INDEX TABLE CHARACTER COUNT 0034

ENTRY H.O.P.	SUBRT NAME	SET I.D.	FIELD SIZE	REL H.O. FIELD POS	FIELD NAME	ARGUMENT LENGTH	FUNCTION LENGTH	C.I. NAME
00005	FXXXS	X	0012	0046	DEST	12	30	CMFLX

3-50

LOGICAL RECORD 9 ELIMINATION INFORMATION TABLE CHARACTER COUNT 0062

TABLE ENTRY	TYPE 1		TYPE 2		TYPE 3	
ENTRY H.O.P.	RECORD ID RFL L.O.P.	FIELD NAME	CROSS INDEX NAME	FIELD NAME	RECORD ID LNG	FIELD NAME
00005	0030	DEPDT	CMFLX	DEST	30	FLITE

SUBROUTINE
SUBIS

LOGICAL RECORD 9 GEOGRAPHIC OPERATION CODE INFORMATION

ENTRY H.O.P.	SUBRT NAME	OP CODE	SUBRT NAME	OP CODE	SUBRT NAME	OP CODE	SUBRT NAME	OP CODE	SUBRT NAME	OP CODE	SUBRT NAME	OP CODE	SUBRT NAME	OP CODE
00039	SUBXS	V	SUBXS	E										

FFT SIZE OF 02950.

Figure 3-22

k. File Structuring Error Comments With Explanations. The following nine pages list all of the possible error comments file structuring may print on the 1403 printer. The list is in alphabetical order for easier reference. Explanations are provided where it is felt they are necessary or helpful. It should be noted that the comments are much easier to interpret when they accompany the input card listing of the card causing the error than they may appear to be in this compilation where there is no input card listing or other supporting data to help clarify the error.

ADVISORY MESSAGE	(Accompanies any of the five-error messages which prevent the structuring of an FFT. These messages may or may not be significant.)
ADD EDIT, SUB AND TAB CARDS MUST PRECEDE THE FIELD-GROUP CARDS	(EDIT, SUB or TAB card(s) out of place. See figure 3-11A.)
COL. 14 INVALID	(Illegal entry type on ELIM card, must be a 1 in col. 14.)
COL. 26 INVALID	(Illegal entry type on ELIM card, must be a 2 in col. 26.)
COL. 45 INVALID	(Illegal entry type on ELIM card, must be a 2 or 3 in col. 45.)
COL. 54 NOT BLANK	(ELIM card with one type 2 entry card probably has one or more extra columns punched.)
COL. 64 INVALID	(Illegal entry type on ELIM card, must be a 3 in col. 64.)
COMMA MISSING	(Comma missing after source ID or after input function on FIELD or GROUP card.)
CONTROL FIELDS MUST BE THE FIRST FIELDS DEFINED	(First field on FIELD card was not a control field.)
CONTROL FIELD/GROUP LIMIT IS 30 CHARACTERS	(Record ID length exceeds 30 characters.)

CROSS-INDEX NAME DOES NOT CORRESPOND TO THIS FFT	(The first four characters of the cross-index name in columns 2 through 5 of the INDEX card should be the same as the first four characters of the file name.)
CROSS-INDEX RECORD (LR8) NOT PRESENT	(The FFT indicates no cross-index table for this file exists; therefore, the ELIM card is meaningless.)
DOUBLY DEFINED EDIT FIELD ID	(Two or more EDIT cards have the same name in columns 2 - 6.)
DOUBLY DEFINED MNEMONIC	(Two fields/groups in the same file cannot be assigned the same name.)
DOUBLY DEFINED SUBROUTINE ID	(Two or more SUB/TAB cards have the same name in columns 2 - 6.)
EXCEEDS FILE MAINTENANCE EXTERNAL FORMAT CARD INPUT LIMITATION	(The size of this field/group exceeds 79 characters. It cannot be maintained by external format input data. This is an ADVISORY message.)
EXCEEDS SOP LIMIT FOR MOVING TO LITERAL	(Size of field/group exceeds 52 characters. This imposes certain (minor) restrictions on the output processing program. This is an ADVISORY message.)
FFT ALREADY IN LIBRARY. IT WILL BE CHANGED IF JOB IS SUCCESSFUL	(CREATE mode specified, but FFT for file name specified already exists. This is an ADVISORY message.)
FFT NOT IN LIBRARY. CHECK FILE ID. IF CORRECT RE-ENTER IN CREATE MODE	(CHANGE mode specified, but file name specified does not exist, or at least has no FFT.)

FFT SIZE OF xxxxx EXCEEDS LIMIT OF 11000.
FIELD CARD ILLEGAL HERE (Field card in input deck was out of sequence. See figure 3-11A.)
FIELDS HAVE DIFFERENT SET ID's (Fields in different sets cannot be defined as a group.)
FIELDS HAVE DIFFERENT TYPE ID's (Fields with different field type ID's cannot be defined as a group.)
FIELDS OUT OF SEQUENCE (Fields listed on GROUP cards must be in the same sequence as the FIELD cards are placed in the deck.)
FIELD TO BE EDITED EXCEEDS CAPACITY OF EDIT FIELD (Inconsistency between information on FIELD card and corresponding information on EDIT card.)
FILE ID. MUST END IN A OR H (The fifth position of the file name is other than A or H.)
FLD. SIZE SHOULD EQ ARG. LNG. (Field size must equal argument length unless a conversion subroutine beginning with F is used.)
FUNCTION LENGTH OF INDEX DOES NOT CORRESPOND TO LR2 CTL. FLD. (Function (record ID) length specified on INDEX card is not equal to the length specified for that record ID in logical record 2 of the FFT.)
FUNCTION REQUIRES SUBSCRIPT (The output function specified is a table or subroutine, which has multiple output sizes subscripted. Subscript the output function to match the desired subscripted output size of the SUB or TAB card.)

GROUP CARD ILLEGAL HERE	(GROUP card is out of sequence in input deck. Refer to "Allowable Card Sequence.")
GROUP CARD OUT OF PLACE	(Group card out of place in relationship to fields it is attempting to group.)
HAS SAME ID AS EDIT FIELD	(A subroutine or table cannot have the same name as an edit word.)
HAS SAME ID AS SUBROUTINE	(Edit name is the same as some subroutine or table name and must be changed.)
ID MUST END IN S	(All subroutine names specified must end in the character "S.")
ILLEGAL CARD TYPE	(When card type was looked up in the card type table, no match was found. Could also result from improper use of BYPASS card.)
ILLEGAL FIELD SIZE	(Field size must be in range of 1 to 910 characters.)
ILLEGAL LOGIC ID	(Logic ID entry in column 20 of FIELD card must be a "blank," an "N," an "S," and "I," or a "T." No other type entry is acceptable.)
ILLEGAL MNEMONIC	(Name given has embedded blanks or is less than 2 characters long, or is otherwise illegal.)
ILLEGAL SUBSCRIPT	(Subscript mispunched.)
ILLEGAL TYPE ID	(Type ID must be C for control field, X for fixed field, or number 1 through 8 for periodic sets. No entry other than the above is acceptable in column 18 of the FIELD card.)

INCORRECT NUMBER OF INPUT	SOURCES. (If input source was a "$," there can be no other input sources listed on FIELD or GROUP card. If input source was not a "$," at least input sources "1" and "2" must be coded on the FIELD or GROUP card.)
INPUT EDIT FIELD HAS ALREADY	BEEN USED AS OUTPUT. (The same edit field cannot be used for both input and output editing.)
INVALID CROSS-INDEX NAME(S).	(The cross-index names in ELIM card must have same first 4 characters as file name. The 5th must be X or Y. If two cross-index names, one must end in X, the other Y.)
JOB CARD OUT OF ORDER.	(FSJOB card out of order. Refer to "Allowable Card Sequence.")
JOB NOT COMPLETED BECAUSE OF	DIAGNOSED ERRORS. (Should be self-explanatory, based upon other error printouts.)
JOB SCRAPPED.	(Job abandoned. Reason may or may not be obvious from previous error printout.)
LOGIC MUST BE THE SAME FOR	ALL FIELDS OF THE SAME SET. (All fields belonging to the same periodic set must carry the same logic mode entry in column 20 of the FIELD card.)
LR 3 OVERFLOW.	(More than 299 elements (fields/groups) specified which exceeds limit. The program maintained fields such as RECCT, PSCTn, VSCTL, PSSQn count as part of the 299.)
LR 4 OVERFLOW.	(Too many periodic sets specified. Limit is eight.)
LR 5 OVERFLOW.	(Too many and/or too lengthy input edit fields. Reduce.)

LR 6 OVERFLOW.	(Too many and/or too lengthy output labels. Reduce.)
LR 7 OVERFLOW.	(Too many and/or too lengthy output edit fields. Reduce.)
MAXIMUM NUMBER OF FIELDS PER GROUP EXCEEDED.	(Maximum number of fields that can be defined as a group is 20.)
MISSING ASTERISK.	(Asterisk not punched (or mispunched) on FIELD or GROUP card.)
MISSING BLANK OR COMMA.	(On SUB or TAB card.)
MISSING BLANK OR #.	(On SUB or TAB card.)
MISSING #.	(On SUB or TAB card. NOTE: # sign may be printed as a = or + on the printer.)
MISSING # ON SUBSCRIPT.	(Subscript on FIELD or GROUP card was not preceded or followed by a #. NOTE. The # may be printed as a = or + on the printer.)
MISSING @ ON EDIT.	(@ may be printed as - or > or '.)
MISSING @ ON EDIT OR LABEL.	(@ may be printed as - or > or '.)
MISSING @ ON LABEL.	(@ sign was omitted or mispunched on FIELD or GROUP card. NOTE: @ may be printed as a - or> or '.)
MODE TYPE IN ERROR.	(Something other than "CREATE" or "CHANGE" in columns 14-19 of FSJOB card.)
MORE THAN THREE CONTINUATION CARDS.	(Maximum allowed is three.)
NO MORE THAN EIGHT FILES MAY BE CREATED IN ONE RUN.	(Self-explanatory.)
NO MORE THAN FIFTEEN FILES MAY BE STRUCTURED IN ONE RUN.	(Self-explanatory.)

NO SUBSCRIPTS ON FUNCTION AS DEFINED.	(The output function specified should not be subscripted.)
NOT IN LIBRARY.	(Subroutine or table not on FFS library. This is an ADVISORY message.)
OUTPUT EDIT FIELD HAS ALREADY BEEN USED AS INPUT.	(The same edit field cannot be used for both input and output editing.)
OUTPUT OF INPUT FUNCTION SMALLER THAN FIELD.	(Field or group size as shown on FIELD or GROUP card is larger than the output size of the corresponding input conversion subroutine table or edit word shown on the SUB/TAB or EDIT card.)
OUTPUT SIZE IN ERROR.	(Refers to output size on SUB or TAB card.)
SUBROUTINE TOO LARGE.	(Size of subroutine exceeds 9999 characters.)
PERIODIC SET ID OUT OF SEQUENCE.	(FIELD card defining periodic field is either mispunched or out of sequence.)
PRECEDING X OR Y INDEX CD. INVALID.	(The previous INDEX card is either out of sequence, doesn't end in an X or a Y, or there are two X cards or two Y cards.)
RECORD ID DOES NOT EQ NO. OF CHAR. IN LR2.	(The record ID length specified in the ELIM card is not equal to the length specified for that record ID in logical record 2 of the FFT.)
REL LOP OF RECORD ID NOT EQ TO OR LS THAN RECORD ID LNG.	(The relative low position specified for the elimination field/group entry in the ELIM card is greater than the record ID length specified.)

SEQUENCE ERROR.	(Refers to sequence numbers in columns 73 - 80. This does not prevent the structuring of an FFT.)
SET ID OUT OF ORDER.	(FIELD card in input deck was either mispunched or out of sequence.)
SIZE OF FIELD EXCEEDS SYSTEM LIMITATION.	(Field size exceeds 910 characters.)
SIZE OF SUBROUTINE TOO SMALL.	(Size of subroutine is less than 60 characters.)
SOURCE ID OUT OF ORDER.	(Input sources listed on FIELD or GROUP card must be in numerical sequence.)
SUBSCRIPT NOT DEFINED.	(If the reference to a function is subscripted, that function (subroutine or table) must also be subscripted.)
SUBROUTINE MUST BEGIN WITH F OR R.	(Subroutine specified on card, columns 38 - 42 must begin with F or R.)
SUBROUTINE OR TABLE * (name) * DEFINED BUT NOT USED.	(If it is defined, it must be used.)
SUBROUTINE TABLE OVERFLOW.	(Excessive number of subroutines defined.)
THIS FILE OUT OF ALPHABETIC SEQUENCE WITH PRECEDING JOB.	(Jobs in a run must be sequenced in alphabetic order by file ID.)
TOO LARGE TO BE QUERIED EXCEPT WITH OVP OPERATOR.	(Field exceeds 52 or 56 characters in length. May only be queried with the OVP geographical polygon operator. This is an ADVISORY message.)

TOO MANY INPUT SOURCES.	(The maximum number of input sources for each field is 9.)
TOO MANY SUBSCRIPTS.	(A maximum of nine subscripts are allowed.)
TYPE 1 FIELD NAME NOT PRESENT IN LR3.	(The field name specified for the elimination field entry in the ELIM card does not exist in logical record 3 of the FFT.)
TYPE 3 FIELD NAME NOT PRESENT IN LR3.	(The record ID specified in the ELIM card does not exist in logical record 3 of the FFT.)
UNDEFINED FIELD ID.	(Name of field (or group) entered on GROUP or INDEX card had not previously been defined on a FIELD (or GROUP) card.)
UNDEFINED INPUT FUNCTION.	(Function was not listed in edit table which was built from information provided on EDIT card.)
UNDEFINED OUTPUT FUNCTION.	(Function was not listed in edit table which was built from information provided on EDIT card.)
UNEXPLAINED ERROR - TRYING TO READ BEYOND ENDFS CARD.	(Error cannot be diagnosed. Recheck input and resubmit job. If error persists, seek assistance from system maintenance programmer personnel.)
VSET CARD ILLEGAL HERE.	(VSET card was out of sequence in input deck. Refer to "Allowable Card Sequence.")
nnnn IS SIZE OF SUBROUTINE.	(The size specified was incorrect. The file structuring phase has used the correct size. <u>This is an ADVISORY message</u>.)

3-2. FILE STRUCTURING SUMMARY: File structuring (FS) is the initial phase of file generation.

a. Input to FS (from the user) consists of a deck of cards defining the structure of the file.

b. Output from FS for each file structured consists of:

(1) A listing of the input deck with messages noting any diagnosed errors.

(2) The file format table (in three forms - printed, tape, and cards).

(3) If CREATE mode is used, a dummy file tape is produced.

c. List of quantitative limitations on data elements in the 1410 FFS:

(1) Up to 299 field/groups can be defined within a file record, of any file. (This number includes the program maintained fields.)

(2) 599 is the maximum number of periodic subsets within any given periodic set.

(3) Up to 8 periodic sets can be defined within a file record of any file.

(4) 30 is the maximum number of characters that can comprise the record ID.

(5) Maximum number of characters allowed in any field or group used as:

(a) A data value (parameter) for retrieval is 56.

(b) A conditional value (parameter) for output processing is 52.

(c) Other than specified in 1 or 2 is 910.

(6) Maximum number of characters allowed in any data record is 5400.

(7) Optimum data element placement and grouping must be determined by the file designer, after taking all factors into consideration.

(8) Variable set data has limitations which may make its use not entirely desirable in certain situations.

d. FS Input Cards - There are thirteen types of input cards acceptable to FS:

FSJOB	SUB	FIELD	INDEX
ENDFS	TAB	GROUP	ELIM
BYPASS	EDIT	VSET	GEOOP
			*(comments)

FSJOB - Begins each file structuring job, supplies file name, specifies mode.

ENDFS - Terminates file structuring.

BYPASS - Causes FS to be immediately bypassed and file revision to be called in for execution.

SUB/TAB - Allows FS to check for the existence of the specified subroutine or table and its size. Also specifies the output size(s) of the subroutine or table.

EDIT - Specifies edit control words to be used for input and/or output editing of file data. Editing may be used to insert most 1410 system characters:

Between data characters.

As a prefix to data characters.

As a suffix to data characters.

Any combination of the above three insertions. In addition, editing may be used to:

Suppress zeros to the left of significant digits.

Selective insertion of the credit symbol, comma, minus sign, asterisk, dollar sign, and decimal.

FIELD - Specifies the field name, size, type, set membership, logic mode (if periodic, input source ID(s) and function(s), output function, and label. The sequence of FIELD cards determines the order of the fields in the file.

GROUP - Specifies the group name and the fields constituting the group. Also specifies the group input source ID(s) and function(s), output function, and label. Immediately follows the FIELD cards which it groups.

VSET - Specifies the name of the variable set, the number of variable set data characters to be output per line, and the label.

INDEX - Specifies the name of the element by which to cross index the file, its size, location, and set membership. It specifies the FxxxS subroutine name. The length of the record ID and the argument length are also specified.

ELIM - Specifies the name and location of the element by which cross-index table functions (record IDs) may be eliminated from consideration for retrieval purposes. It specifies other associated data concerning the name, etc., of the cross index, record ID, and element by which the file is cross indexed. It also specifies the subroutine(s) to convert field name for cross index processor.

GEOOP - Specifies the subroutine name and the geographic operation code(s) to replace the general geographic operator at file retrieval.

* - Allows any comments desired to be inserted in the printed listing output by FS.

Correct card sequence is critical for input cards to FS.

The File Format Table is made up of nine logical records.

1 - ID Record

2 - Detail Information Table

3 - Field/Group Lookup Table

4 - Set ID Table

5 - The Input Edit Record

6 - Label Table

7 - The Output Edit Record

8 - The Cross-Index Table

9 - Elimination Information and Geographic Operator Code Table

e. File Structuring Error Comments. Advisory messages and SEQUENCE ERROR do not prevent the structuring of the FFT. Other messages do.

3-3. FILE REVISION:

a. General.

(1) When generating a new file only the file structuring phase of file generation need be executed. When restructuring (reformatting) the data records of an existing file, after FS is executed it calls in file revision

(FR), the second phase of file generation. FR requires FS to have structured a new file format table (FFT) for the file which is one of the inputs FR uses to reformat the data file. If the new FFT is presently available on tape (from some previous job) and it is now desired to execute file revision without first having to execute FS, the first card after the "MON$$ EXEQ FG" card should be a "BYPASS FS" card. This causes FS to immediately call in FR, bypassing FS completely.

(2) The changes to the data file which may be accomplished by FR include:

(a) Rearrangement of the sequence of fields or periodic sets in the file records.

(b) Addition of new fields or periodic sets to file records.

(c) Deletion of fields or periodic sets from file records.

(d) Changing the size of fields in file records.

(e) Changing the name of fields in the file record, or changing the name of the file itself.

More than one of the above changes can be accomplished concurrently on an element of a file record. For example, a field may have both a name and a size change during one execution of file revision.

(3) It must be emphasized that file revision operates on only the smallest data elements in a file, i.e., fields. Groups and subsets are ignored in the FR program logic. If the fields comprising a group or subset are revised, the groups and subsets are thereby automatically revised.

(4) File revision should not be confused with file maintenance whose purpose is to change the data content of a file, not the format; whereas the purpose of file revision is to change the format of a file not its data content.

(5) To ADD a field to an existing fixed or periodic set in the file revision sense, means to increase the size of each file record by the insertion of a blank field. Thus, the fixed set or each applicable periodic subset in each of the records of the revised file contains the allotted space for the new field. If the new FFT calls for the addition of a new periodic set, the file revision program inserts a new periodic set control field containing blanks into the fixed set of each revised file record. The actual data for the new fields or periodic sets must be inserted during the execution of file maintenance.

(6) To DELETE a fixed or periodic field in the file revision sense is to decrease the size of each file record by removing all traces of the field. Thus the fixed set or each applicable periodic subset in each of the records of the revised file would no longer provide space for the deleted

field. If the new FFT does not specify a periodic set that existed in the old file, the file revision program removes all of the subsets in applicable periodic set as well as the periodic set control field from each record of the revised data file.

(7) Any change in the size of the file is due to the insertion or deletion of fields or a change in the size of existing fields. A revised file may be physically larger or smaller than the original file depending upon the nature of the changes. However, the data content of the fields retained in the revised file, will be exactly the same as before revision.

(8) Before FR begins the actual revision of a file, it develops a set of machine language instructions for each field and periodic set to be changed. The information necessary for the development of these instruction sets is derived from two sources. The first source is a comparison of the field entries in the old and new FFT's. The second source is information contained in the FR change cards and FR control cards which are input to FR.

(9) Figure 3-23 gives a general picture of the overall operation of the file revision phase of file generation, showing inputs, outputs, and the major operations performed.

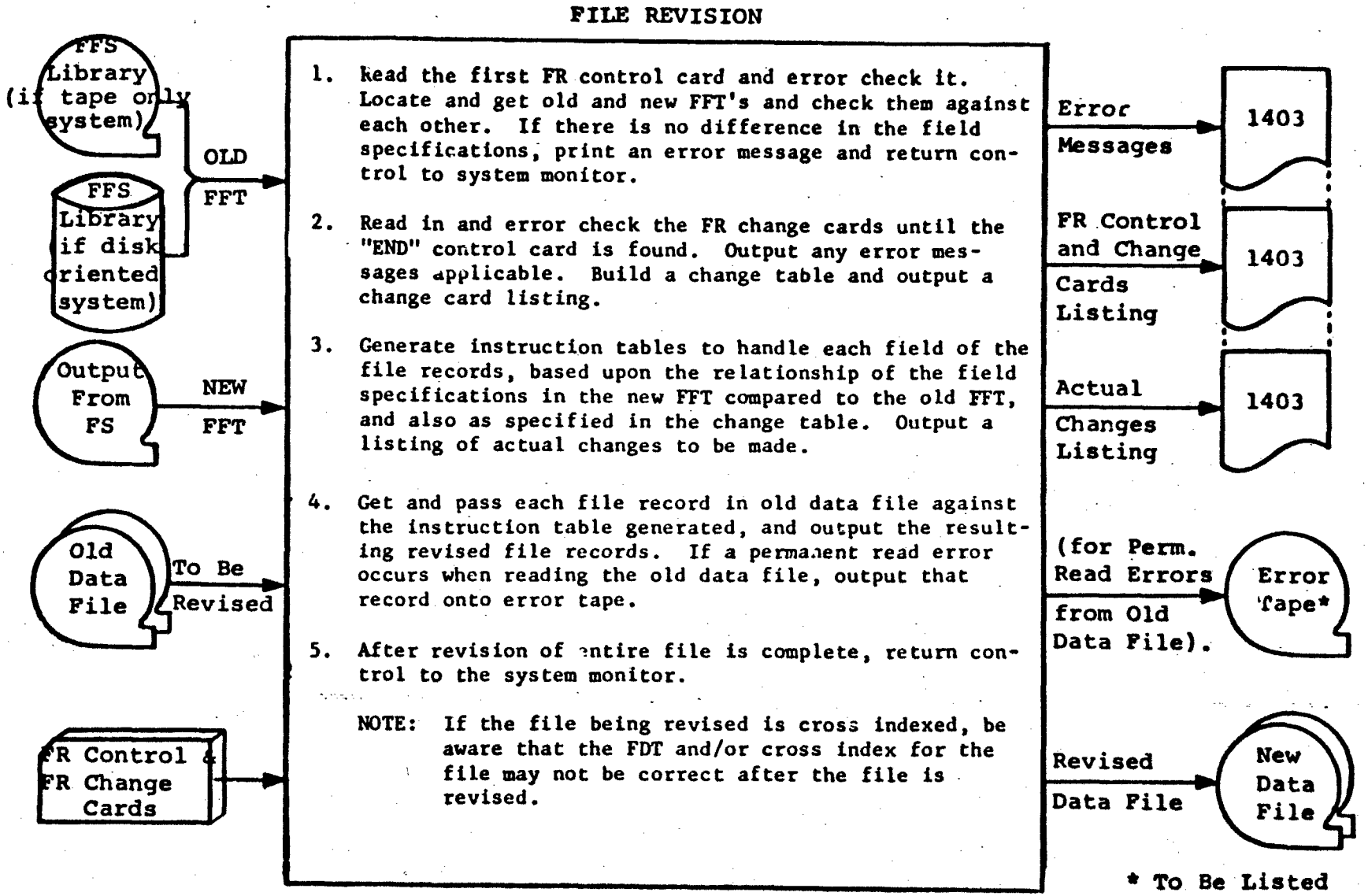


Figure 3-23. General Operation Of The File Revision Phase.

* To Be Listed by OP For Recovery of Data.

b. FR Input Card Specifications.

(1) FR Control Cards. The required FR control cards are coded and formatted as follows:

First Card of The Change Deck (File Name Card)

If no name change is desired, put present file name in both fields.

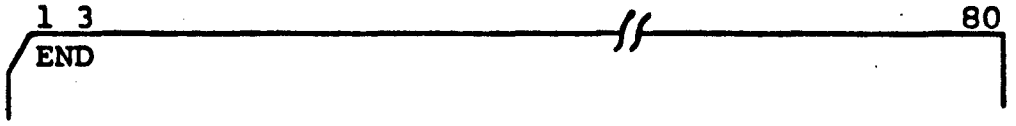
Column 1 An asterisk.

Columns 2-6 The File Name BEFORE revision.

Columns 7-11 The File Name AFTER revision.

Columns 12-80 Blanks.

Last Card Of The Change Deck (End Card)



Columns 1-3 The word END

Columns 4-80 Blanks.

Both of these cards are required in every File Revision run, even if the File ID does not change.

(2) FR Change Cards.

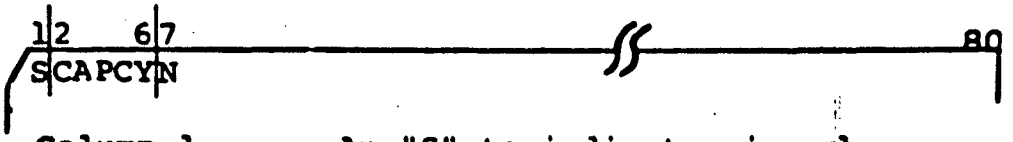
(a) The information from the FR change cards is used to construct a "change table." The change table contains the specific information concerning which fields and/or periodic sets are to be revised, and in what manner the changes are to be accomplished. This table contains information pertaining to a change of field name, and/or size, the addition of a periodic or variable set, and a change of sequence of periodic sets.

(b) The addition or deletion of a field, the actual amount of change in size, and/or the rearrangement of the sequence of fields is determined by comparing parts of the old FFT against corresponding parts of the new FFT. The information obtained by the comparison of the old and new FFTs is used in conjunction with the change table to generate an "instruction table." The instruction table is used to perform the actual revision of the old file into the new file, on a field by field basis.

(c) There may be up to five different types of FR change cards input to FR. There must be one FR control card (file name card) preceding the FR change cards and one FR control card (END card) following the FR change cards. (It should be noted that some revisions will not require the use of any FR change cards, in which case only the two FR control cards are input to FR.) Each of the five types of FR change cards is made unique and identified by the character punched in column one. This single character indicates the type of revision specified by the FR change card.

(d) The FR change cards are coded and formatted as follows:

A Change of Field Size (S)



Column 1 — An "S" to indicate size change.

Columns 2-6 — The 2 to 5 character field name, left justified.

Column 7 — An "A" or "N" to indicate Alphameric or Numeric field.

Columns 8-80 — Blanks.

NOTE: No indication is made as to the actual change in field size. That is determined by comparing the field size specified in logical record number 2 of the new FFT against that of the old FFT.

A Change of Field Name (N)

Column 1 — An "N" to indicate name change.

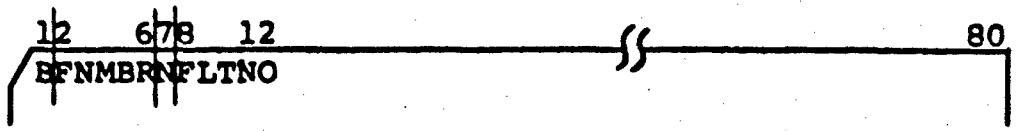
Columns 2-6 — "Old" 2 to 5 character field name, left justified.

Column 7 — Blank

Columns 8-12 — "New" 2 to 5 character field name, left justified.

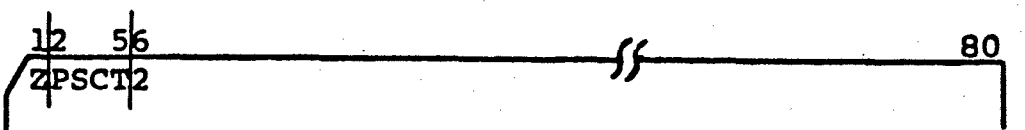
Columns 13-80 — Blanks.

A Change of Field Name and Size (B)



Column 1	A "B" to indicate both a name and a size change.
Columns 2-6	"Old" 2 to 5 character field name, left justified.
Column 7	An "A" or "N" to indicate an alphameric or numeric field.
Columns 8-12	"New" 2 to 5 character field name, left justified.
Columns 13-80	Blanks.

Addition Of A Periodic Set or A Variable Set (Z)



Column 1	A "Z" to indicate the addition of a Periodic or Variable Set.
Columns 2-5	The letters PSCT
Column 6	The Periodic Set ID number. (Use 9 for Variable Set.)
Columns 7-80	Blanks.

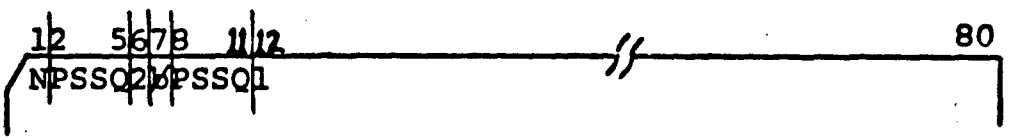
Change of Periodic Set ID Number (C) (Two Cards Required)

First Card

Column 1	A "C" to indicate a change in Periodic Set ID number.
Columns 2-5	The letters PSCT
Column 6	The Periodic Set ID number before revision.
Column 7	Blank.
Columns 8-11	The letters PSCT
Column 12	The Periodic Set ID number after revision.
Columns 13-80	Blanks.

The second card is a Name Change Card (N) having the same type of format already given for the Field Name Change Card (N). To logically follow the above card, this card must change the name PSSQ2 to PSSQ1.

Second Card



Column 1	An "N" to indicate name change.	
Columns 2-5	The letters PSSQ	Old Name
Column 6	"Old" PSSQ number.	
Column 7	Blank.	
Columns 8-11	The letters PSSQ	New Name
Column 12	"New" PSSQ number.	
Columns 13-80	Blanks.	

c. Use of "Interim" FFT's,

(1) Because the file revison program will only handle fields, some revisions must make use of a so-called "interim" FFT; e.g.: assume that the "old" FFT for a file specifies a field entry as shown below:

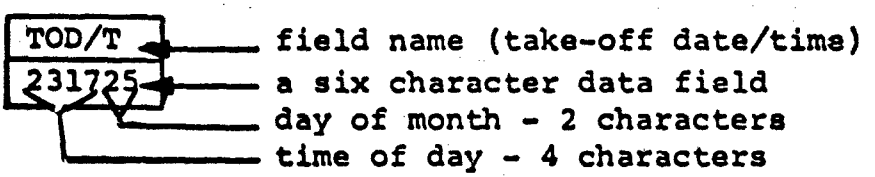


Figure 3-24. Original Configuration of "Take-off Date/Time" Data

(2) Assume, further, that the user desires to rearrange the data in this field and at the same time, wishes to raise TOD/T to the group level:

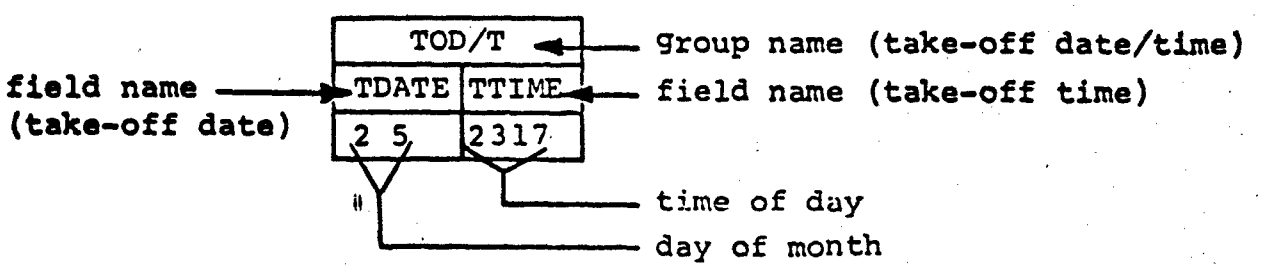


Figure 3-25. Desired Configuration of "Take-off Date/Time" Data

(3) If the user's present file were revised using the old and new FFTs (as inputs to the file revision program) erroneous results would occur. The TOD/T entry in the new FFT would be ignored, since it is a group entry, and the resulting revision logic for the TDATE and TTIME fields would be such as to add them as new fields. The data in TOD/T would NOT be transferred from the old file to the revised file. The TDATE and TTIME fields on the revised file would contain only blanks.

(4) To solve this problem, the user must restructure his present (old) FFT, raising TOD/T to the group level (inserting TTIME and TDATE as fields in the group) and place this "interim" FFT on the FFS Relocatable Execution Library prior to the execution of file revision. With this having been done, file revision treats the "interim" FFT as the "old" FFT. (Refer to figure 3-26.)

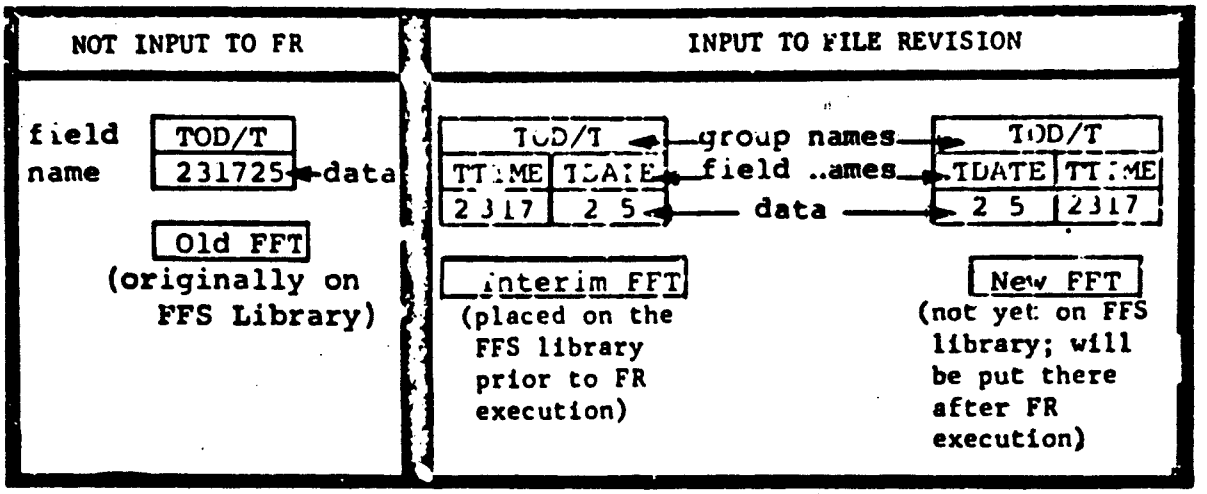


Figure 3-26. Relationship of Old, Interim, and New FFTs

(5) The resulting revision logic for the TTIME and TDATE fields would be a change in relative location only. The data in the TTIME and TDATE fields would be transferred and rearranged in the revised file since they are described in the "interim" old FFT.

(6) The use of interim FFTs is very effective whenever the user wishes to revise only a portion or sections of a field. For example, if the user wished simply to rearrange the data in the original TOD/T field in the manner described above, without raising TOD/T to the group level, he would still use exactly the same approach outlined above. Then just put the original FFT (with TOD/T at the field level) back on the FFS Relocatable Execution Library by a library update run.

(7) By using a combination of "interim" FFTs, it is possible to truncate fields, to build fields from portions of other fields, to insert blanks or zeros in any position of a field, and finally, to create a new FFS data file from portions of existing FFS data files.

d. File Revision Examples, Using The CMFLA File. A hypothetical preliminary version of the sample file CMFLA will be revised in two following examples, so that at the completion of example #2, the format will have been made identical to the CMFLA file created in the file structuring run example. Each example will show:

A graphic of the file's format before revision.

The FS input card listing used to structure the old FFT.

A graphic of the file's format after revision.

The FS input card listing used to structure the new FFT.

The FR change cards required.

Example #1

This example illustrates:

A field size change,

a field name change,

a field name and size change,

the addition of a field, and

the deletion of a field.

Figure 3-28 is a graphic of the file before revision. Figures 3-29 through 3-31 show the FS input cards used to structure the old FFT (describing the file before revision).

The following changes are made in example #1:

The field CAPCY is increased from 2 to 3 characters.

The field FNMBR is decreased from 4 to 3 characters and its name is changed to FLTNO.

The field AVGAL is renamed ALTTD and its relative location is changed.

A three-character field named ONBRD is added between FUELL and the new location of ALTTD.

The fields AOD and AOT which form the group AOD/T are deleted.

In addition to the old and new FFTs the following cards are necessary to accomplish the above revision:

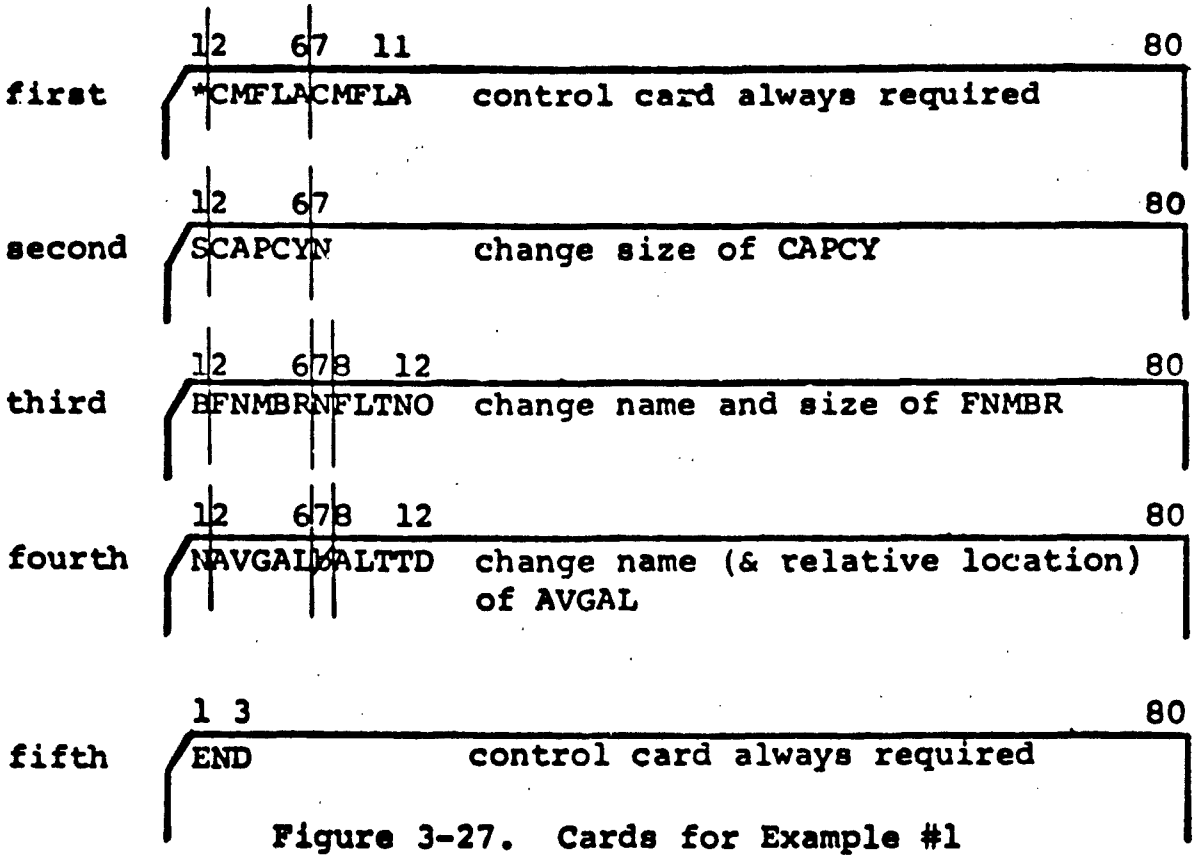


Figure 3-27. Cards for Example #1

(1) Notice that no FR change cards are required to add or delete a field. File revison accomplishes this by comparing the old and new FFTs against each other.

(2) Figure 3-32 is a graphic of the file after revision by example #1. Figures 3-33 through 3-35 show the FS input cards used to structure the new FFT (describing the revised file.)

FIXED SET

RECCT	FLITE						DEST	FNMBR	ACTYP	FLTYP	CAPCY	PSCT1	PSCT2
			DEPDT										
	ANAME	ORGIN	DYEAR	DMNTH	DDATE	DHOUR							
4	8	12	2	2	2	4	12	4	4	1	2	8	8

PERIODIC SET 1

PSSQn	RNGRP			
	RNWYL	RNWYW	RNWYS	RNWYC
3	5	4	4	1

PERIODIC SET 2

PSSQn	LORIG	LTERM	GEREF		LXOFY	AVGAL	TOD/T		AOD/T		FUELL	GROWT
			OCOOR	TCOOR			TDATE	TTIME	AOD	AOT		
3	12	12	15	15	2	3	2	4	2	4	3	4

Figure 3-28. Preliminary Version of the CMFLA File Before Revision by Example #1.

```
FILE  STRUCTURING                          ***  CREATE  CMFLA  ***        COPY 02  F 02        DATE  18NOV     PAGE  01

                                                                                                                001CMFLA
      FSJOB CREATE,CMFLA,DATE,NOOK,LST1                                                                         ADVISORY MESSAGE
                  FFT ALREADY IN LIBRARY. IT WILL BE CHANGED IN LIB. UPDATE RUN.

          * FILE NAME   COMMERCIAL FLIGHTS FILE                                                    002CMFLA
          * FILE ID     CMFLA                                                                     003CMFLA
          ************************************************************************              004CMFLA
          *** THE PURPOSE OF THUS FS JOB IS TO STRUCTURE A PRELIMINARY VERSION ***              005CMFLA
          *** OF THE CMFLA SAMPLE FILE. TO SHOW HOW FILE REVISION CAN BE USED. ***              006CMFLA
          ************************************************************************              007CMFLA
          *** THIS REPRESENTS THE OLD FFT FOR FILE REVISION EXAMPLE NUMBER ONE ***              008CMFLA
          ************************************************************************              009CMFLA

FNMBR EDIT  * 0 *                                                                               010CMFLA

DEDCO EDIT  *  +DEG+  +MIN+  +SEC+ ,+   +DEG+  +MIN+  +SEC+ .*                                  011CMFLA

GFREF EDIT  *  D  M  S+ ,   D  M  S+ ,  D  M  S+ ,   D  M  S+ .*                                012CMFLA

LXOFY EDIT  *LEG+ +OF+ *                    USED FOR LXOFY                                      013CMFLA

IFLTS TAB   0061.001                                                                            014CMFLA
                  NOT IN LIBRARY.                                                               ADVISORY MESSAGE

HUNDS SUB   0061.007       USED FOR ALTID, FUELL, AND GROWT                                     015CMFLA
                  NOT IN LIBRARY.                                                               ADVISORY MESSAGE

OPTCS SUB   0259.005=1=,016=2=                                                                  016CMFLA
                  NOT IN LIBRARY.                                                               ADVISORY MESSAGE

          ************************************************************************              017CMFLA
          * THE FOLLOWING COMPRISE THE FIXED SET.                                               018CMFLA

ANAME FIELD 008,C, .*$,ALPHA,*      .*AIRLINE*                                                  019CMFLA

ORGIN FIELD 012,C, .*$,ALPHA,*      .*FLT ORIGIN*                                               020CMFLA

DYEAR FIELD 002,C, .*$,NUMER,*      .*YEAR*                                                     021CMFLA

DMNTH FIELD 002,C, .*$,NUMER,*      .*MONTH*                                                    022CMFLA

DDATE FIELD 002,C, .*$,NUMER,*      .*DAY*                                                      023CMFLA

DHOUR FIELD 004,C, .*$,NUMER,*      .*HOUR*                                                     024CMFLA

DEPDT GROUP DYEAR,DMNTH,DDATE,DHOUR,*$,NUMER,*OPTCS=2=,*DEPARTURE DATE TIME*                   025CMFLA
                                                                                                SEQUENCE ERROR.
```


Figure 3-29

```
FILE  STRUCTURING                              *** CREATE CMFLA ***        COPY 02 OF 02        DATE 18NOV      PAGE 02

FLTF GROUP ANAME,ORGIN,DEPDT,*$,ALPHA,*      ,*FLIGHT*
CREDT FIELD 006,X, ,*$,NUMER,*      ,*CREATE DATE*
CHGDT FIELD 006,X, ,*$,NUMER,*      ,*CHANGE DATE*
DEST  FIELD 012,X, ,*$,ALPHA,*      ,*DESTINATION*                                                    028CMFLA
FNMBR FIELD 003,X, ,*$,FNMBR,*      ,*FLT*                                                            029CMFLA
ACTYP FIELD 004,X, ,*$,ALPHA,*      ,*A/C TYPE*                                                       030CMFLA
FLTYP FIELD 001,X, ,*1,ALPHA,2,ALPHA,3,IFLTS,*OPTCS=1=,*FLIGHT TYPE*                                  031CMFLA
CAPCY FIELD 003,X, ,*$,NUMER,*      ,*CAPACITY*                                                       032CMFLA
          ********** THE FOLLOWING COMPRISE PERIODIC SET ONE **********                  033CMFLA
RNWYL FIELD 005,1,S,*$,NUMER,*      ,*RNWY LENGTH*                                                    034CMFLA
RNWYW FIELD 004,1,S,*$,NUMER,*      ,*RNWY WIDTH*                                                     035CMFLA
RNWYS FIELD 004,1,S,*$,ALPHA,*      ,*RNWY SURFACE*                                                   036CMFLA
RNWYC FIELD 001,1,S,*$,ALPHA,*      ,*RNWY CONDITION*                                                 037CMFLA
RNGRP GROUP RNWYL,RNWYW,RNWYS,RNWYC,*$,ALPHA,*      ,*RUNWAY DATA*                                038CMFLA
PSET1 GROUP PSS01,RNGRP,*$,ALPHA,*      ,*PERIODIC SET ONE*                                           039CMFLA
          ********** THE FOLLOWING COMPRISE PERIODIC SET TWO **********                  040CMFLA
LORIG FIELD 012,2,S,*$,ALPHA,*      ,*LEG ORIGIN*                                                     041CMFLA
LTERM FIELD 012,2,S,*$,ALPHA,*      ,*LEG TERMINATION*                                                042CMFLA
OCOOR FIELD 015,2,S,*$,ALPHA,*DEDCO,*LEG ORIGIN COORD*                                                043CMFLA
TCOOR FIELD 015,2,S,*$,ALPHA,*DEDCO,*LEG TERM. COORD*                                                 044CMFLA
GEREF GROUP OCOOR,TCOOR,*$,ALPHA,*GEREF,*LEG ORIGIN COORDINATES*                                      045CMFLA
LXOFY FIELD 002,2,S,*$,NUMER,*LXOFY,*LEG X OF Y*                                                      046CMFLA
AVGAL FIELD 003,2,S,*$,NUMER,*HUNDS,*AVG ALT*                                                         047CMFLA
```


Figure 3-30

```
FILE  STRUCTURING                                    *** CREATE CMFLA ***        COPY 02 OF 02        DATE 18NOV     PAGE 03

                                                                                                                  04ACMFLA
TDATE FIELD 002.2.S.*$.NUMER.*      .*TO DATE*                                                                    049CMFLA
TTIME FIELD 004.2.S.*$.NUMER.*      .*TO TIME*                                                                    050CMFLA
TDD/T GROUP TDATE.TTIME.*$.NUMER.*      .*TO DT/TIME*
                                                                                          051CMFLA
      = *                                                                                                         052CMFLA
ADD   FIELD 002.2.S.*$.NUMER.*      .*ARRIVAL DATE*                                                               053CMFLA
ADT   FIELD 004.2.S.*$.NUMER.*      .*ARRIVAL TIME*                                                               054CMFLA
ADD/T GROUP ADD  .ADT  .*$.NUMER.*      .*ARR DT/TIME*                                                            055CMFLA
FUELL FIELD 003.2.S.*$.NUMER.*HUNDS.*FUEL LOADED*                                                                 056CMFLA
GRDWT FIELD 004.2.S.*$.NUMER.*HUNDS.*TAKEOFF WEUGHT*                                                              057CMFLA
PSET2 GROUP PSS02.LORIG.LTERM.GEREF.LXDFY.AVGAL.TDD/T.ADD/T.FUELL.GRDWT.*$.ALPHA.*      .*PERIODIC SET TWO*      SEQUENCE ERROR.
                        EXCEEDS FILE MAINTENANCE EXTERNAL FORMAT CARD INPUT LIMITATION.                           ADVISORY MESSAGE
                        TOO LARGE TO BE QUERIED EXCEPT WITH OVP OPERATOR.                                         ADVISORY MESSAGE
                        EXCEEDS SDP LIMIT FOR MOVING TO LITERAL.                                                  ADVISORY MESSAGE
                                                                                          059CMFLA
      *                                                                                                           060CMFLA
CMFLX INDEX DEST .X.0046.0012.12.30.FXXXS                                                                         ADVISORY MESS GE
                        NOT IN LIBRARY.                                                                           061CMFLA
      FLIN  IDEPDT.0030 2SUBIS.DFST .CMFLX 3FLITE.30
                                                                                          061CMFLA                SEQUENCE ERROR.
      *                                                                                                           062CMFLA
      GEODP SUBXS.V SUBXS.F
                                                                                          063CMFLA
      *                                                                                                           063CMFLA
      ENDFS
```


Figure 3-31

FIXED SET

RECCT	FLITE						DEST	FLTNO	ACTYP	FLTYP	CAPCY	PSCT1	PSCT2
	ANAME	ORGIN	DEPDT										
			DYEAR	DMNTH	DDATE	DHOUR							
4	8	12	2	2	2	4	12	3	4	1	3	8	8

PERIODIC SET 1

PSSQn	RNGRP			
	RNWAY	RNWD	RNSUF	RNCON
3	5	4	4	1

PERIODIC SET 2

PSSQn	LORIG	LTERM	GEREF		LXOFY	TOD/T		ALTTD	ONBRD	FUELL	GROWT
			OCOOR	TCOOR		TDATE	TTIME				
3	12	12	15	15	2	2	4	3	3	3	4

Figure 3-32. Preliminary Version of the CMFLA File <u>After</u> Revision By Example #1, and <u>Prior To</u> Revision by Example #2.

```
FILE  STRUCTURING                                  ***  CHANGE  CMFLA  ***        COPY 01 OF 02        DATE  24NOV     PAGE  01

                                                                                                                        001CMFLA
      FSJOB CHANGE.CMFLA.DATF.NOOK.LST1
                                                                                                                        002CMFLA
          * FILE NAME   COMMERCIAL FLIGHTS FILE                                                                         003CMFLA
          * FILE ID     CMFLA                                                                                          004CMFLA
          ************************************************************************                                      005CMFLA
          *** THE PURPOSE OF THIS FS JOB IS TO STRUCTURE A PRELIMINARY VERSION ***                                      006CMFLA
          *** OF THE CMFLA SAMPLE FILE, TO SHOW HOW FILE REVISION CAN BE USED. ***                                      007CMFLA
          ************************************************************************                                      008CMFLA
          *** THIS REPRESENTS THE NEW FFT FOR EX. 1, AND THE OLD FFT FOR EX.2. ***                                      009CMFLA
          ************************************************************************
                                                                                                                        010CMFLA
FLTNO EDIT  * 0 *  SUPPRESSES UP TO 2 LEADING ZEROS.  USED FOR FLTNO.
                                                                                                                        011CMFLA
OFOCO EDIT  *  +DEG+  +MIN+  +SEC+ .+  +DEG+  +MIN+  +SEC+ .*
                                                                                                                        012CMFLA
GEREF EDIT  *  D  M  S+ .  D  M  S+ .  D  M  S+ .  D  M  S+ .*
                                                                                                                        013CMFLA
LXOFY EDIT  *LEG+ +OF+ *                    USED FOR LXOFY
                                                                                                                        014CMFLA
EFITS TAB   0061.001                                                                                          ADVISORY MESSAGE
                    NOT IN LIBRARY.
                                                                                                                        015CMFLA
HUNDS SUB   0061.007      USED FOR ALTTO, FUELL, AND GRONT                                                    ADVISORY MESSAGE
                    NOT IN LIBRARY.
                                                                                                                        016CMFLA
OPTCS SUB   0259.005=1=,016=2=                                                                                ADVISORY MESSAGE
                    NOT IN LIBRARY.
                                                                                                                        017CMFLA
          ************************************************************************                                      018CMFLA
          * THE FOLLOWING COMPRISE THE FIXED SET.
                                                                                                                        019CMFLA
ANAME FIELD 009,C, .*$,ALPHA,*      .*AIRLINE*
                                                                                                                        020CMFLA
ORGIN FIELD 012,C, .*$,ALPHA,*      .*FLT ORIGIN*
                                                                                                                        021CMFLA
OYEAR FIELD 002,C, .*$,NUMER,*      .*YEAR*
                                                                                                                        022CMFLA
OMNTH FIELD 002,C, .*$,NUMER,*      .*MONTH*
                                                                                                                        023CMFLA
ODATE FIELD 002,C, .*$,NUMER,*      .*DAY*
                                                                                                                        024CMFLA
OHOUR FIELD 004,C, .*$,NUMER,*      .*HOUR*
                                                                                                                        025CMFLA
DEPDT GROUP OYEAR,OMNTH,ODATE,OHOUR,*$,NUMER,*OPTCS=2=,*DEPARTURE DATE TIME*                                  SEQUENCE ERROR.
```

3-81

DIAM 65-9-1

Figure 3-33

```
FLITE GROUP ANAME.ORGIN.DEPDT.*$.ALPHA.*      .'FLIGHT'
CREDT FIELD 006.X. .*$.NUMER.*      .'CREATE DATE'
CHGDT FIELD 006.X. .*$.NUMER.*      .'CHANGE DATE'
DEST  FIELD 012.X. .*$.ALPHA.*      .'DESTINATION'                                  028CMFLA
FLTNO FIELD 003.X. .*$.FLTNO.*      .'FLT'                                          029CMFLA
ACTYP FIELD 004.X. .*$.ALPHA.*      .'A/C TYPE'                                     030CMFLA
FLTYP FIELD 001.X. .*1.ALPHA.2.ALPHA.3.(FLTS.*OPTCS=1=.'FLIGHT TYPE'                031CMFLA
CAPCY FIELD 003.X. .*$.NUMER.*      .'CAPACITY'                                     032CMFLA
          ********** THE FOLLOWING COMPRISE PERIODIC SET ONE **********           033CMFLA
RNWYL FIELD 005.1.S.*$.NUMER.*      .'RNWY LENGTH'                                  034CMFLA
RNWYW FIELD 004.1.S.*$.NUMER.*      .'RNWY WIDTH'                                   035CMFLA
RNWYS FIELD 004.1.S.*$.ALPHA.*      .'RNWY SURFACE'                                 036CMFLA
RNWYC FIELD 001.1.S.*$.ALPHA.*      .'RNWY CONDITION'                               037CMFLA
RNGRP GROUP RNWYL.RNWYW.RNWYS.RNWYC.*$.ALPHA.*      .'RUNWAY DATA'              038CMFLA
PSET1 GROUP PSSQ1.RNGRP.*$.ALPHA.*      .'PERIODIC SET ONE'                     039CMFLA
          ********** THE FOLLOWING COMPRISE PERIODIC SET TWO **********           040CMFLA
LORIG FIELD 012.2.S.*$.ALPHA.*      .'LEG ORIGIN'                               041CMFLA
LTERM FIELD 012.2.S.*$.ALPHA.*      .'LEG TERMINATION'                          042CMFLA
OCOOR FIELD 015.2.S.*$.ALPHA.*DEDCO.'LEG ORIGIN COORD'                          043CMFLA
TCOOR FIELD 015.2.S.*$.ALPHA.*DEDCO.'LEG TERM. COORD'                           044CMFLA
GEREF GROUP OCOOR.TCOOR.*$.ALPHA.*GEREF.'LEG ORIGIN COORDINATES'                045CMFLA
LXOFY FIELD 002.2.S.*$.NUMER.*LXOFY.'LEG X OF Y'                                046CMFLA
TDATE FIELD 002.2.S.*$.NUMER.*      .'TO DATE'                                  048CMFLA
```


Figure 3-34

```
FILE  STRUCTURING                                  *** CHANGE  CMFLA  ***        COPY 02 OF 02          DATE  18NOV     PAGE  03

TTIME FIELD 004.2.S.*$.NUMER.*      .*TO TIME*                                                                          049CMFLA
TOD/T GROUP TDATE.TTIME.*$.NUMER.*      .*TO DT/TIME*                                                                   050CMFLA
          *                                                                                  051CMFLA
ALTTD FIELD 003.2.S.*$.NUMER.*HUNDS.*AVG ALT*                                                                         052CMFLA
ONBRD FIELD 003.2.S.*$.NUMER.*      .*ON BOARD*                                                                       053CMFLA
FUELL FIELD 003.2.S.*$.NUMER.*HUNDS.*FUEL LOADED*                                                                     055CMFLA
GROWT FIELD 004.2.S.*$.NUMER.*HUNDS.*TAKEOFF WEIGHT*                                                                  056CMFLA
PSET2 GROUP PSS02.LORIG.LTERM.GEREF.LXDFY.TOD/T.ALTTD.ONBRD.FUELL.GROWT.*$.ALPHA.*     .*PERIODIC SET TWO*          058CMFLA
                        TOO LARGE TO BE QUERIED EXCEPT WITH OVP OPERATOR.                                    SEQUENCE ERROR.
                        EXCEEDS SDP LIMIT FOR MOVING TO LITERAL.                                             ADVISORY MESSAGE
                                                                                                             ADVISORY MESSAGE
          *                                                                                  059CMFLA
CMFLX INDEX DEST .X.0046.0012.12.30.FXXXS                                                                             060CMFLA
                        NOT IN LIBRARY.                                                                      ADVISORY MESSAGE
      FLIN  *DEPDT.CC30 2SUBLS.DEST .CMFLX *FLITE.30                                                                  061CMFLA
          *                                                                                  061CMFLA
                                                                                                             SEQUENCE ERROR.
      GROUP SUBXS.V SUBXS.F                                                                                           062CMFLA
          *                                                                                  063CMFLA
      ENDFS                                                                                                           063CMFLA
```


Figure 3-35

Example #2

The resulting revised file of example #1 will be further revised by example #2. Example #2 illustrates:

The addition of a periodic set,

the deletion of a periodic set,

changing a periodic set ID number,

the addition of two periodic fields and grouping them, and

the addition of a variable set.

Figure 3-32 is a graphic of the file before revision by example #2 (and after example #1). Figures 3-33 through 3-35 show the FS input cards used to structure the FFT which is the old FFT for example #2 (and the new FFT for example #1).

The following changes are made in example #2:

The current periodic set #1 is deleted,

the existing periodic set #2 ID number is changed from 2 to 1,

two fields (grouped) are added to the periodic set that is changed from 2 to 1,

a new periodic set #2 is added, and

a variable set is added.

In addition to the old and new FFTs, the following cards are required to accomplish the above changes:

In addition to the old and new FFTs, the following cards are required to accomplish the above changes:

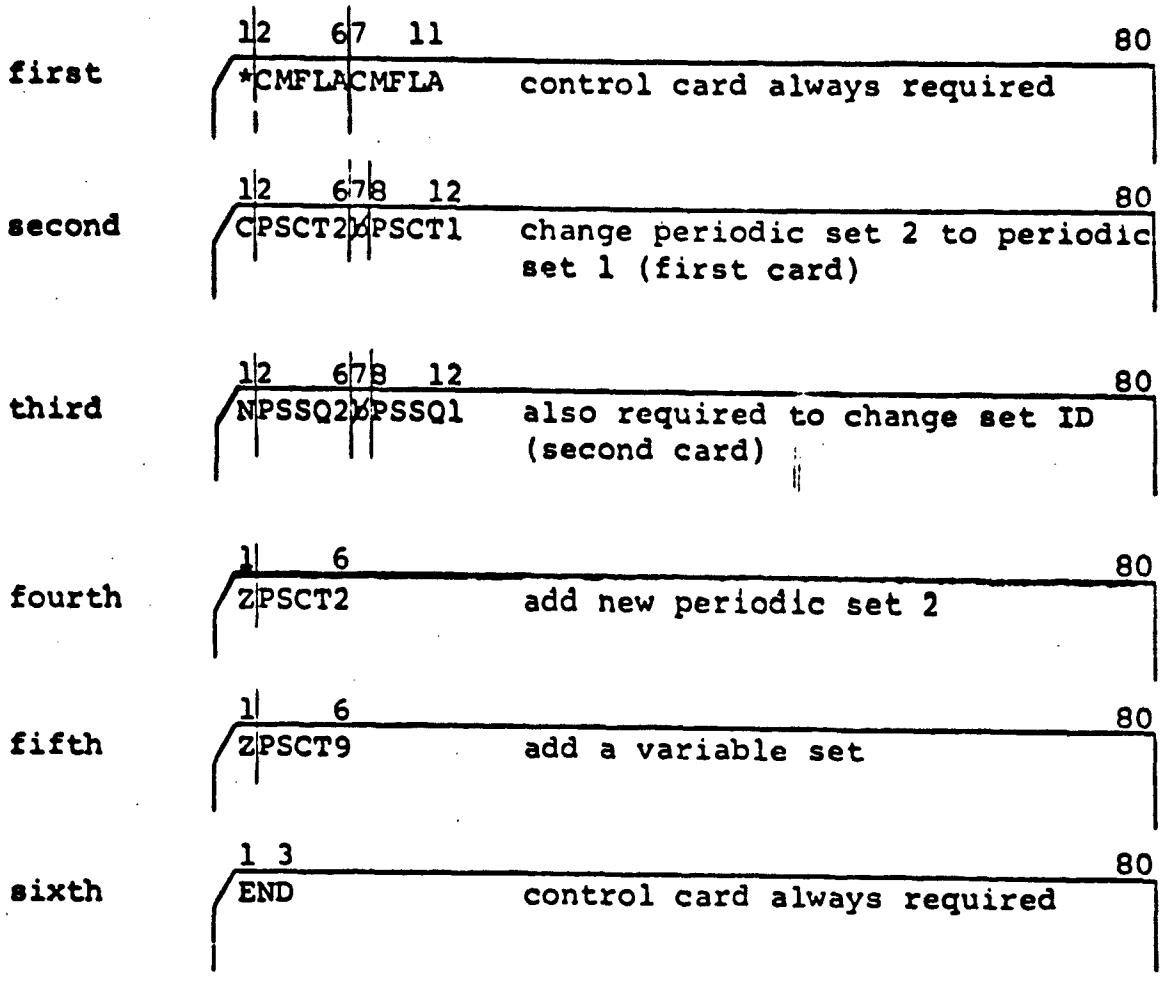


Figure 3-36. Cards for Example #2

Notice that an FR change card is not required to delete a periodic set or to add the new fields. File revision accomplishes this by comparing the new and old FFTs against each other.

Figure 3-38 is a graphic of the file after revision by example #2. (Notice that this is exactly the same format of the CMFLA file illustrated by figure 2-2 in Section Two.) The FS input cards used to structure the new FFT for example #2 are shown in figures 3-39 through 3-41. They are identical to the FS input cards used earlier in the CMFLA file structuring example, except that CHANGE mode is used here, to cause execution of file revision after FS.

FIXED SET

RECCT	FLITE						DEST	FLTNO	ACTYP	FLTYP	CAPCY	PSCT1	PSCT2	VSCTL
	ANAME	ORGIN	DEPDT											
			DYEAR	DMNTH	DDATE	DHOUR								
4	8	12	2	2	2	4	12	3	4	1	3	8	8	8

DIAM 65-9-1

PERIODIC SET 1

PSSQn	LORIG	LTERM	GEREF		LXOFY	TOD/T		LD/T		ALTTD	ONBRD	FUELL	GROWT
			OCOOR	TCOOR		TDATE	TTIME	LDATE	LTIME				
3	12	12	15	15	2	2	4	2	4	3	3	3	4

PERIODIC SET 2

PSSQn	LOCAN	MAINT								RLOSS		
		NCOMM	FRAME	ELSYS	HYSYS	ENGIN	LGEAR	ENMNT	RPLCD	OTHER	MEALS	ROOMS	PNLTY
3	12	1	1	1	1	1	1	1	1	1	3	3	3

VARIABLE SET

REMRK

NOTE: The revised file does not contain any data in periodic set 2 or the variable set. Data for them can only be supplied by a File Maintenance run.

Figure 3-38. The CMFLA File After Revision By Both Examples #1 and #2.

```
      FSJOB CHANGE,CMFLA,DATE,MODK,LST1

          * FILE NAME   COMMERCIAL FLIGHTS FILE                                  002CMFLA
          * FILE ID     CMFLA                                                    003CMFLA
          * OBJECTIVE   THE CMFLA FILE IS A SAMPLE FILE USED THROUGHOUT THE      004CMFLA
          *** THE PURPOSE OF THIS FS JOB IS TO STRUCTURE A FILE FORMAT TABLE ***  005CMFLA
          *** REPRESENTING THE OLD FFT FOR FILE REVISION EXAMPLE NUMBER TWO. ***  006CMFLA
          ********************************************************************** 007CMFLA
          *                                                                      008CMFLA
          * THIS FS INPUT DECK IS IDENTICAL TO THE FS INPUT DECK USED TO STRUCTURE 009CMFLA
          * THE CMFLA  FFT EARLIER IN SECTION THREE  TO ILLUSTRATE THE USE OF FILE 010CMFLA
          * STRUCTURING.  THE ONLY DIFFERENCE IS THE USE OF  CHANGE  MODE IN THIS  011CMFLA
          * FS JOB TO CAUSE FILE REVISION TO BE EXECUTED AFTER STRUCTURING THE FFT 12CMFLA
          ********************************************************************** 013CMFLA
          *                                                                      014CMFLA
          * THE FOLLOWING IS USED FOR INPUT EDITING                              015CMFLA

FLTNO EDIT  * 0 *  SUPPRESSES UP TO 2 LEADING ZEROS.  USED FOR FLTNO.                016CMFLA

          * THE FOLLOWING ARE USED FOR OUTPUT EDITING                            017CMFLA

PNLTY EDIT  *$   .*                                                                  018CMFLA

OEDCO EDIT  *  +DEG+  +MIN'  +SEC+ .+   +DEG+  +MIN+  +SEC+ .*                      019CMFLA

          *                              USED FOR OCOOR AND TCOOR.             020CMFLA

GEREF EDIT  *  D  M  S+ .  D  M  S+ .  D  M  S+ .  D  M  S+ .*                       021CMFLA

          *                                                    FOR GEREF.      022CMFLA

LXOFY EDIT  *LEG+ +OF+ *                  USED FOR LXOFY                             023CMFLA

          ********************************************************************** 024CMFLA
          * THE FOLLOWING IS USED FOR THE INPUT CONVERSION OF THE FLTYP FIELD.  * 025CMFLA
          * IT CONVERTS THE NUMBERS 1 THRU 5 TO P, C, D, T, OR X RESPECTIVELY.  * 026CMFLA

IFLTS TAB   0061.001                                                                 027CMFLA
                    NOT IN LIBRARY.                                          ADVISORY MESSAGE

          *THE FOLLOWING OUTPUT CONVERSION SUBROUTINE PLACES A COMMA JUST PRIOR TO 028CMFLA
          *THE LOW ORDER DIGIT OF THE DATA AND THEN ADDS TWO TRAILING ZEROS TO IT. 029CMFLA

MUNDS SUB   0061.007        USED FOR ALTID, FUELL, AND GROWT                         030CMFLA
                    NOT IN LIBRARY.                                          ADVISORY MESSAGE
```


Figure 3-39

```
        * THE FOLLOWING SUBROUTINE IS USED FOR OUTPUTTING EITHER THE FLTYP FIELD      031CNFLA
        * OR THE DEPOT GROUP.  IT CONVERTS THE DATA IN FLTYP FROM EITHER P TO        032CNFLA
        * PASSG, C TO CARGO, D TO DEDHD, F TO CHART, OR S TO OTHER.  IF THE DATA     033CNFLA
        * FROM DEPOT IS BEING CONVERTED, IT OUTPUTS THE SCHED DEPT. DATE/TIME IN     034CNFLA
        * THE FORM OF  DD MMM YYYY HHMM,  E.G. 19 NOV 1965 1723.                     035CNFLA

OPTCS SUB   0259,005*1*,016*2*                                                       036CNFLA
                  NOT IN LIBRARY.                                                ADVISORY MESSAGE

        ********************************************************************       037CNFLA
        * THE FOLLOWING COMPRISE THE FIXED SET.                                     038CNFLA

ANAME FIELD 004,C, ,0$,ALPHA,0      ,'AIRLINE'                                     039CNFLA

ORGIN FIELD 012,C, ,0$,ALPHA,0      ,'FLT ORIGIN'                                  040CNFLA

DYEAR FIELD 002,F, ,0$,NUMER,0      ,'YEAR'                                        041CNFLA

DMNTH FIELD 002,C, ,0$,NUMER,0      ,'MONTH'                                       042CNFLA

DDATE FIELD 002,C, ,0$,NUMER,0      ,'DAY'                                         043CNFLA

DHOUR FIELD 004,C, ,0$,NUMER,0      ,'HOUR'                                        044CNFLA

DEPOT GROUP DYEAR,DMNTH,DDATE,DHOUR,0$,NUMER,0OPTCS*2*,'DEPARTURE DATE TIME'       045CNFLA
                                                                              SEQUENCE FIELD.

FLTE GROUP ANAME,ORGIN,DEPOT,0$,ALPHA,0      ,'FLIGHT'
CREDT FIELD 006,8, ,0$,NUMER,0      ,'CREATE DATE'

CHGDT FIELD 006,8, ,0$,NUMER,0      ,'CHANGE DATE'

DEST  FIELD 012,8, ,0$,ALPHA,0      ,'DESTINATION'                                 046CNFLA

FLTNO FIELD 003,8, ,0$,FLTNO,0      ,'FLT'                                         047CNFLA

ACTYP FIELD 004,8, ,0$,ALPHA,0      ,'A/C TYPE'                                    048CNFLA

FLTYP FIELD 001,8, ,01,ALPHA,2,ALPHA,3,IFLTS,0OPTCS*1*,'FLIGHT TYPE'               049CNFLA

CAPCY FIELD 003,8, ,0$,NUMER,0      ,'CAPACITY'                                    050CNFLA

        *                                                                          051CNFLA
        *                                                                          052CNFLA
        ********** THE FOLLOWING COMPRISE PERIODIC SET ONE **********             053CNFLA
```

Figure 3-40

```
FILE  STRUCTURING                          *** CHANGE  CMFLA ***        COPY 01 OF 02        DATE  ENDV      PAGE  03

LORIG FIELD 012,1,S,*$,ALPHA,*      ,*LEG ORIGIN*                                                          079CMFLA
LTERM FIELD 012,1,S,*$,ALPHA,*      ,*LEG TERMINATION*                                                     080CMFLA
GCOOR FIELD 015,1,S,*$,ALPHA,*DEGCO,*LEG ORIGIN COORD*                                                     081CMFLA
TCOOR FIELD 015,1,S,*$,ALPHA,*DEGCO,*LEG TERM. COORD*                                                      082CMFLA
GEREF GROUP OCOOR,TCOOR,*$,ALPHA,*GEREF,*LEG ORIGIN COORDINATES*                                           083CMFLA
LXOFY FIELD 002,1,S,*$,NUMER,*LEOFY,*LEG X OF Y*                                                           084CMFLA
TDATE FIELD 002,1,S,*$,NUMER,*      ,*T  DATE*                                                             085CMFLA
TTIME FIELD 004,1,S,*$,NUMER,*      ,*T  TIME*                                                             086CMFLA
TOD/T GROUP TDATE,TTIME,*$,NUMER,*      *TO DT/TIME*                                                       087CMFLA
LDATE FIELD 002,1,S,*$,NUMER,*      ,*L  DATE*                                                             088CMFLA
LTIME FIELD 004,1,S,*$,NUMER,*      ,*LEG TIME*                                                            089CMFLA
LD/T  GROUP LDATE,LTIME,*$,NUMER,*      ,*LOG DT/TIME*                                                     090CMFLA
ALTTD FIELD 003,1,S,*$,NUMER,*HUNDS,*ALTG ALT*                                                             091CMFLA
ONBRD FIELD 003,1,S,*$,NUMER,*      ,*ON BOARD*                                                            092CMFLA
FUELL FIELD 003,1,S,*$,NUMER,*HUNDS,*FUEL LOADED*                                                          093CMFLA
GROWT FIELD 004,1,S,*$,NUMER,*HUNDS,*TAKEOFF WEUGHT*                                                       094CMFLA
        *** THE FOLLOWING CARD CAUSES A NO. 5 TYPE ID ENTRY IN LR2 OF THE FFT **          095CMFLA
PSET1 GROUP PSSOI,LORIG,LTERM,GEREF,LXOFY,TOD/T,LD/T ,ALTTD,ONBRD,FUELL,GROWT,*$,ALPHA,*    ,*PERIODIC SET ONE*      096CMFLA
                                                                                                  SEQUENCE ERROR.
                    EXCEEDS FILE MAINTENANCE EXTERNAL FORMAT CARD INPUT LIMITATION.                ADVISORY MESSAGE
                    TOO LARGE TO BE QUERIED EXCEPT WITH OVP OPERATOR.                              ADVISORY MESSAGE
                    EXCEEDS SORT LIMIT FOR MOVING TO LITERAL.                                      ADVISORY MESSAGE
        *                                                                                 097CMFLA
        *                                                                                 098CMFLA
        ********** THE FOLLOWING COMPRISE PERIODIC SET TWO**********                     099CMFLA
LOCAN FIELD 012,2,S,*$,ALPHA,*      ,* LOCATION*                                                           100CMFLA
```

Figure 3-41

```
NCOMM FIELD 001,2,S,*$,NUMER,*      ,*NAV/COMM*                                                    079CMFLA
FRAME FIELD 001,2,S,*$,NUMER,*      ,*AIRFRAME*                                                    080CMFLA
ELSYS FIELD 001,2,S,*$,NUMER,*      ,*ELEC SYS*                                                    081CMFLA
HYSYS FIELD 001,2,S,*$,NUMER,*      ,*HYDR SYS*                                                    082CMFLA
ENGIN FIELD 001,2,S,*$,NUMER,*      ,*ENGINE*                                                      083CMFLA
LGEAR FIELD 001,2,S,*$,NUMER,*      ,*LDG GEAR*                                                    084CMFLA
ENMNT FIELD 001,2,S,*$,NUMER,*      ,*ENV CNTL*                                                    085CMFLA
RPLCD FIELD 001,2,S,*$,NUMER,*      ,*A/C REPL*                                                    086CMFLA
OTHER FIELD 001,2,S,*$,NUMER,*      ,*OTHER*                                                       087CMFLA
MAINT GROUP NCOMM,FRAME,ELSYS,HYSYS,ENGIN,LGEAR,ENMNT,RPLCD,OTHER,*$,NUMER,*      ,*UNSCH MAINT*  089CMFLA
                                                                                                  SEQUENCE ERROR.

MEALS FIELD 003,2,S,*$,NUMER,*      ,*MEALS PURCH*                                                 090CMFLA
ROOMS FIELD 003,2,S,*$,NUMER,*      ,*PASS BILLET*                                                 091CMFLA
PNLTY FIELD 003,2,S,*$,NUMER,*PNLTY,*FRGHT PNLTY*                                                  092CMFLA
RLOSS GROUP MEALS,ROOMS,PNLTY,*$,NUMER,*      ,*REVENUE LOSS*                                      093CMFLA

        *** THE FOLLOWING CARD CAUSES A NO. 5 TYPE ID ENTRY IN LR2 OF THE FFT **          094CMFLA

PSET2 GROUP PSS02,LOCAN,MAINT,RLOSS,*$,ALPHA,*      ,*SET 2*                                       095CMFLA

        *                                                                                 096CMFLA
        *                                                                                 097CMFLA
        ************** FOLLOWING IS THE VARIABLE SET SPECIFICATION************ *          098CMFLA

REMRK VSET  120,*R E M A R K S   R E M A R K S   R E M A R K S*                                   099CMFLA
                    EXCEEDS FILE MAINTENANCE EXTERNAL FORMAT CARD INPUT LIMITATION.          ADVISORY MESSAGE
                    TOO LARGE TO BE QUERIED EXCEPT WITH OVP OPERATOR.                        ADVISORY MESSAGE
                    EXCEEDS SOP LIMIT FOR MOVING TO LITERAL.                                 ADVISORY MESSAGE

        *                                                                                 100CMFLA
        *                                                                                 101CMFLA
        * THE FOLLOWING CARD SPECIFIES THAT A CROSS INDEX TABLE OF ARGUMENTS AND          102CMFLA
        * FUNCTIONS IS TO BE BUILT AND MAINTAINED FOR THIS FILE. FOR EACH UNIQUE         103CMFLA
        * DATA CONTENT OF THE DEST  FIELD THERE IS AN ARGUMENT. THE FUNCTIONS OF         104CMFLA
        * THE ARGUMENTS ARE THE RECORD IDS OF THE RECORDS HAVING THE ARGUMENT IN         105CMFLA
        * THEIR DEST  FIELD. .                                                          106CMFLA
```

Figure 3-42

```
FILE  STRUCTURING                          *** CHANGE  CMFLA ***        COPY 01 OF 02        DATE  1740V      PAGE  04

CMFLX INDEX DEST ,(,0046,0012,12,3( , XXXS                                                                  107CMFLA
                    NOT IN LIBRARY.                                                                  ADVISORY MESSAGE

          *                                                                          108CMFLA
          * THE FOLLOWING SPECIFICATION ALLOWS THE SELECTIVE ELIMINATION OF RECORD  109CMFLA
          * IDS *FUNCTIONS* IN THE DEST  CROSS INDEX TABLE FROM CONSIDERATION *FOR  110CMFLA
          * RETRIEVAL PURPOSES*, BASED UPON THE DATA CONTENT OF THE DEPOT GROUP.    111CMFLA
          *
      ELIM  1DEPOT,C030 2SUBIS,DEST ,CMFLX 3FLITE,30                                                    112CMFLA

      GROUP SUBXS,V SUBXS,F                                                                             113CMFLA

          *                                                                          114CMFLA

      ENDFS                                                                                             115CMFLA
```


Figure 3-43

e. File Revision Error Comments, With Explanations.

(1) The following two pages list the error and informative messages that file revision may printout on the 1403 printer. The third page lists the explanations of the abbreviations used in the actual changes listing. In addition to those shown, file revision also lists the 80 column card image of any input card in which a format error is detected, followed by: ***ERROR***.

(2) Other items listed by file revision are:

The new and old data file names.

The FR change cards listing.

The changes to actually be made to the data file (actual changes listing). These are listed by field, with a 5 or 10 character abbreviation for what is being done to the field.

CHANGE TABLE BUILT INCORRECTLY - CHECK ANDESCRIPTOR ON SIZE CHANGE CARDS	(Check the Alpha/Numeric descriptor character in column 7 of change card(s) specifying a field size change or both a field size and name change.)
CHANGE TABLE BUILT INCORRECTLY - CHECK CHANGE CARDS	(Either the character in column 1 is not allowable, or it is not the appropriate character based upon the fields in the rest of the card, or the END card is incorrect in column 2 or 3.)
CHECK PSN 1 of 1ST CARD OF REVISION DECK FOR *	(First control card read from change deck did not have an asterisk in column 1. Card sequence may be incorrect or card may be mispunched.)
DATA CHECK OR WRONG LENGTH RECORD ON OUTPUT DATA - MOUNT A NEW SCRATCH TAPE ON B1	(A permanent data check occurred while writing the new revised data file onto tape B1. Job should be resubmitted as is, after insuring a good scratch tape is put on B1.)
DATA CHECK OR WRONG LENGTH RECORD ON NEW FFT FILE	(A permanent read error results when attempting to read in the new FFT from tape. Check the possibility of some incorrect (even parity) reel being used. If not, FS can be re-run to obtain another new FFT tape.)
EITHER FFT FOR FILE TO BE REVISED CANNOT BE FOUND OR FILE NAME IN CONTROL CARD IS WRONG	(Check the file mnemonic in columns 2-6 of the first control card. If correct, insure that a file format table for the subject file is on the FFS System Library.)
EITHER FFT FOR NEW FILE IS NOT ON B5 or FILE NAME IN CONTROL CARD IS WRONG	(Check the file mnemonic in columns 7-11 of the first control card. If correct, the reel containing the new file format table probably wasn't mounted on tape unit B5.)

END OF JOB - FILE REVISION RUN COMPLETE	(Self explanatory.)
JOB HAS BEEN SCRAPPED FOR ABOVE REASON	(Self explanatory. Appears after any FR error comment.)
OLD AND NEW FFT ARE IDENTICAL - EITHER THERE IS NO NEED FOR FILE REVISION OR LIBRARY UPDATE HAS BEEN RUN SINCE FILE STRUCTURING	(Determine which is the case. If the latter and the data file is still in the old format, the FFT on the library must be replaced by the FFT for the unrevised data file.)
OLD FIELD NAME IN CHANGE TABLE IS NOT IN OLD FFT - CHECK CHANGE CARDS AND OLD FFT	(Compare columns 2-6 of change cards against actual field names in old file format table.)
PSCT(n) IS NEITHER IN OLD FFT OR CHANGE TABLE.	(The specified periodic set ID number is in the new FFT. However, it cannot be found in the old FFT, and it was not specified on a change periodic set ID number card or on an addition of a periodic set card. Check any such cards against new FFT for wrong set ID number.)
nnn WLR OR DATA CHECKS WERE ENCOUNTERED ON INPUT DATA FILE. RUN OUTPUT PROGRAM USING TAPE ON CHAN B3 AS INPUT. THIS MUST BE DONE BEFORE LIBRARY UPDATE	(This message results when one or more wrong length record checks or data checks occur while reading the "old" input data file. Any records from the "old" data file that cannot be read without causing one of the above errors are output onto an "FR error tape," (Tape unit B3). It should be obvious that any records put on B3 are in the format specified by the "old" FFT, which is still on the FFS Library while FR is being run. This FR error tape from B3 may be listed by using output processing (OP execution phase) in the "source direct" mode, specifying the (old) file name. This listing of the FR error tape MUST be done before the old FFT on the FFS Library is replaced with the new FFT by a library update run.

Explanation of Abbreviations
Used in Actual Changes Listing

f. The actual changes listing gives the element name and a 5- or 10-character abbreviation for what is being done to that entry. The explanation of these abbreviations follows:

ELEMENT NAME	CHARACTER ABBREVIATION	EXPLANATION
COUNT	RECAL	The character count will be recalculated after the record has been revised.
XXXXX	CHNAM	The field name has been changed
XXXXX	FLAGG	This entry has been flagged as a group or control field.
XXXXX	NALGR	The new alphabetic field is greater.
XXXXX	NALSM	The new alphabetic field is smaller.
XXXXX	NEWFD	This is a new field.
XXXXX	NEWST	This field is part of a new periodic set.
XXXXX	NNMGR	The new numeric field is greater.
XXXXX	NNMSM	The new numeric field is smaller.
XXXXX	NOCHG	No change has been made to this field.
XXXXX XXXXX XXXXX XXXXX	NAMEA NALSM NAMEA NALGR NAMEA NNMSM NAMEA NNMGR	These messages indicate that the field name has been changed as well as the size.
VSETØ	MOVED	This means that the variable set will be moved from the old to the new record.

Figure 3-44. Abbreviations Used in Actual Changes Listing

3-4. FILE REVISION SUMMARY:

a. FR - Purpose is to change the format of a file, not its data contents.

b. BYPASS FS causes file structuring to immediately call in file revision (FR), bypassing FS completely.

c. THE CHANGES to the data file which FR can perform include:

(1) Rearrangement of the sequence of fields or periodic sets in the file records.

(2) Addition of new fields or periodic sets to file records.

(3) Deletion of fields or periodic sets or the variable set from file records.

(4) Changing the size of fields in file records.

(5) Changing the name of fields or changing the name of the file itself.

d. FR OPERATES only on fields. Groups and subsets are ignored in the FR program logic.

e. FR Input Card Specifications.

FR Control Cards

File Name Card	Both required
END Card	in any FR run

FR Change Cards. Each of the five types of FR change cards is made unique and identified by the character punched in column one:

"S" indicates size change.

"N" indicates name change

"B" indicates both a name and a size change.

"Z" indicates the addition of a periodic set or a variable set.

"C" indicates a change in periodic set ID number.

f. Changes Not Specified in Input Cards. The following changes are accomplished by comparing the old and new FFTs with each other and are not specified in the input cards to FR:

Adding a field,

deleting a field,

deleting a pe. ..ic se* or the variable ..et.

changing the relative location of a field within its set,

the actual amount of size change. (An S or B FR change card, only indicates a change is to be made.)

g. File Revision Outputs. The revised data file is the prime output of FR. The following are the different li-tings that may be output on the 1403 printer:

(1) Error comments and informative messages, including the new and old data file names.

(2) The 80 column card image of any input card with a format error.

(3) The FR change cards listing.

(4) Actual changes listing.

File revision may also output an FR error tape containing those records from the old data file which cannot be read without errors.

h. "Interim" FFTs may be useful whenever the user wishes to revise only a portion or sections of a field.

SECTION FOUR

FILE MAINTENANCE

4-1. GENERAL: The creation of new file records and the addition, deletion, or changing of the data content of the file record elements (called updating) is accomplished by file maintenance.

4-2. GENERAL OPERATION OF EACH PHASE OF FILE MAINTENANCE: A general description of each phase of file maintenance has already been given in the latter part of Section Two of this manual. A somewhat more detailed and yet still general description of the operation of each phase is given in the next six illustrations. These figures summarize the general operation of the FM phases as follows:

Figure 4-0 - FM Supervisor.

Figure 4-1 - Input Processor.

Figure 4-2 - FM Sort.

Figure 4-3 - FM Proper.

Figure 4-4 - Cross-Index Updater.

Figure 4-5 - Subset Purger.

FILE MAINTENANCE SUPERVISOR

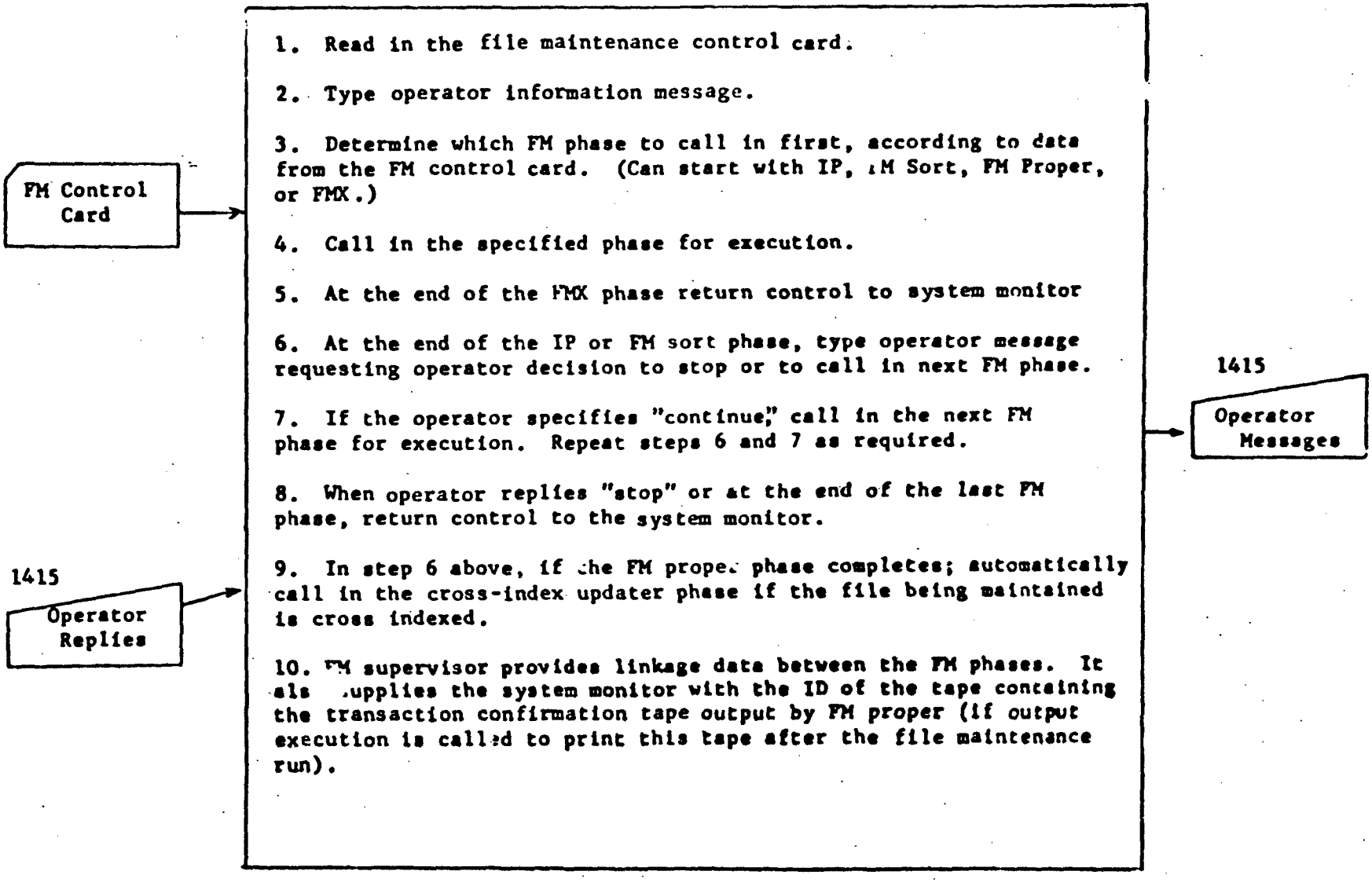


Figure 4-0. General Operation of FM Supervisor.

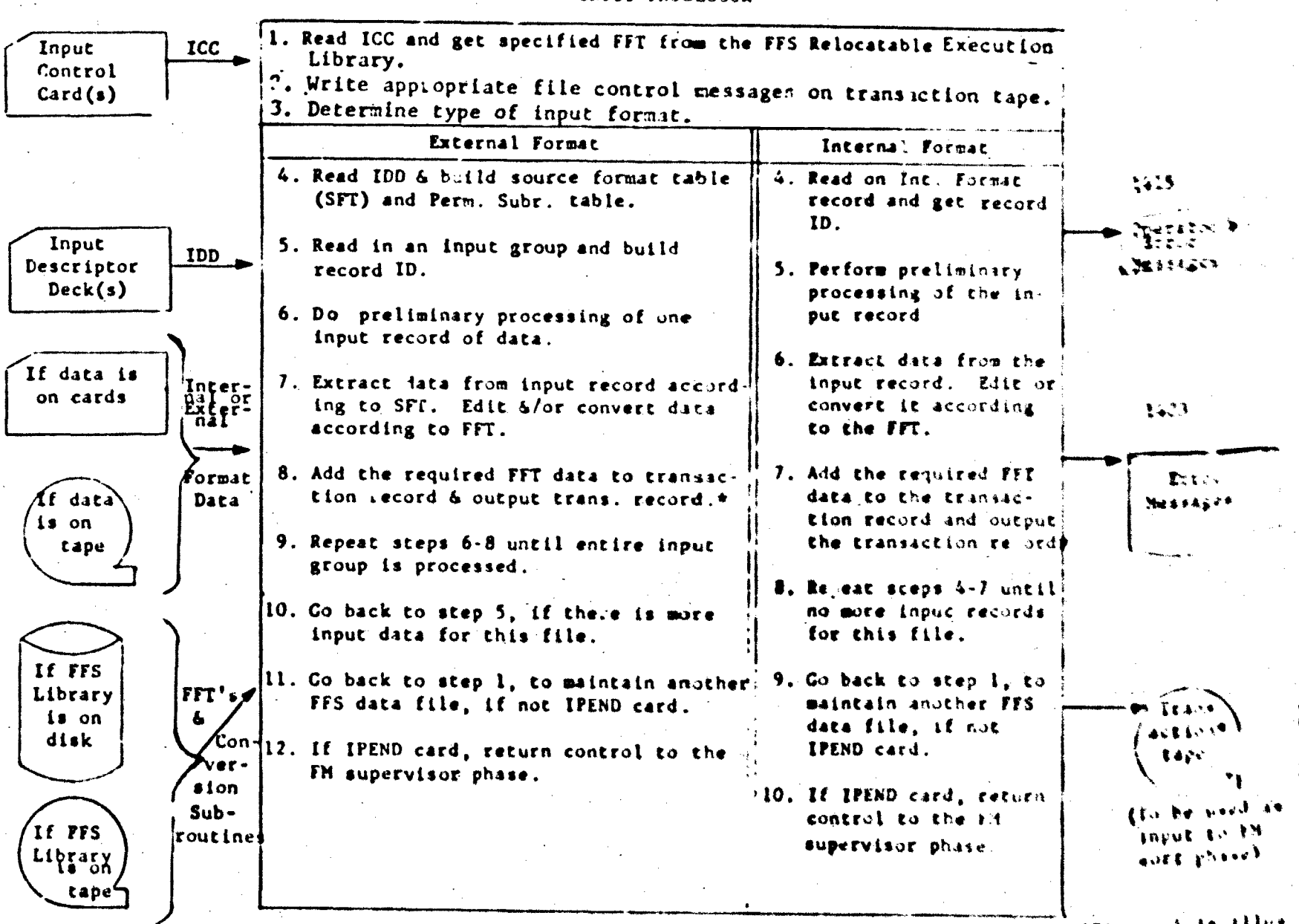


Figure 4-1. General Operation of Input Processing.

*Figure 4-16 illustrates transaction record.

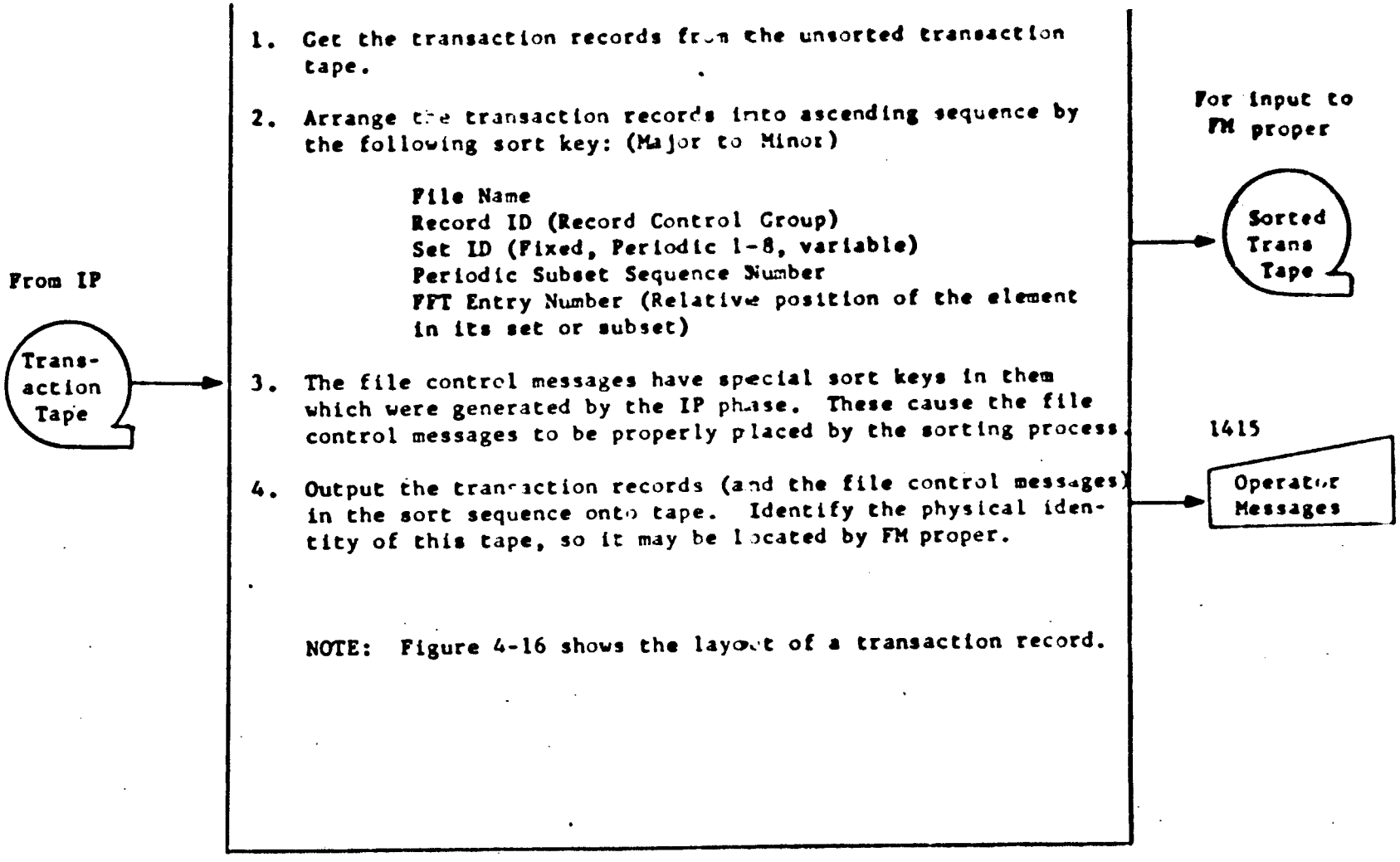


Figure 4-2. General Operation of FM Sort.

FILE MAINTENANCE PROPER

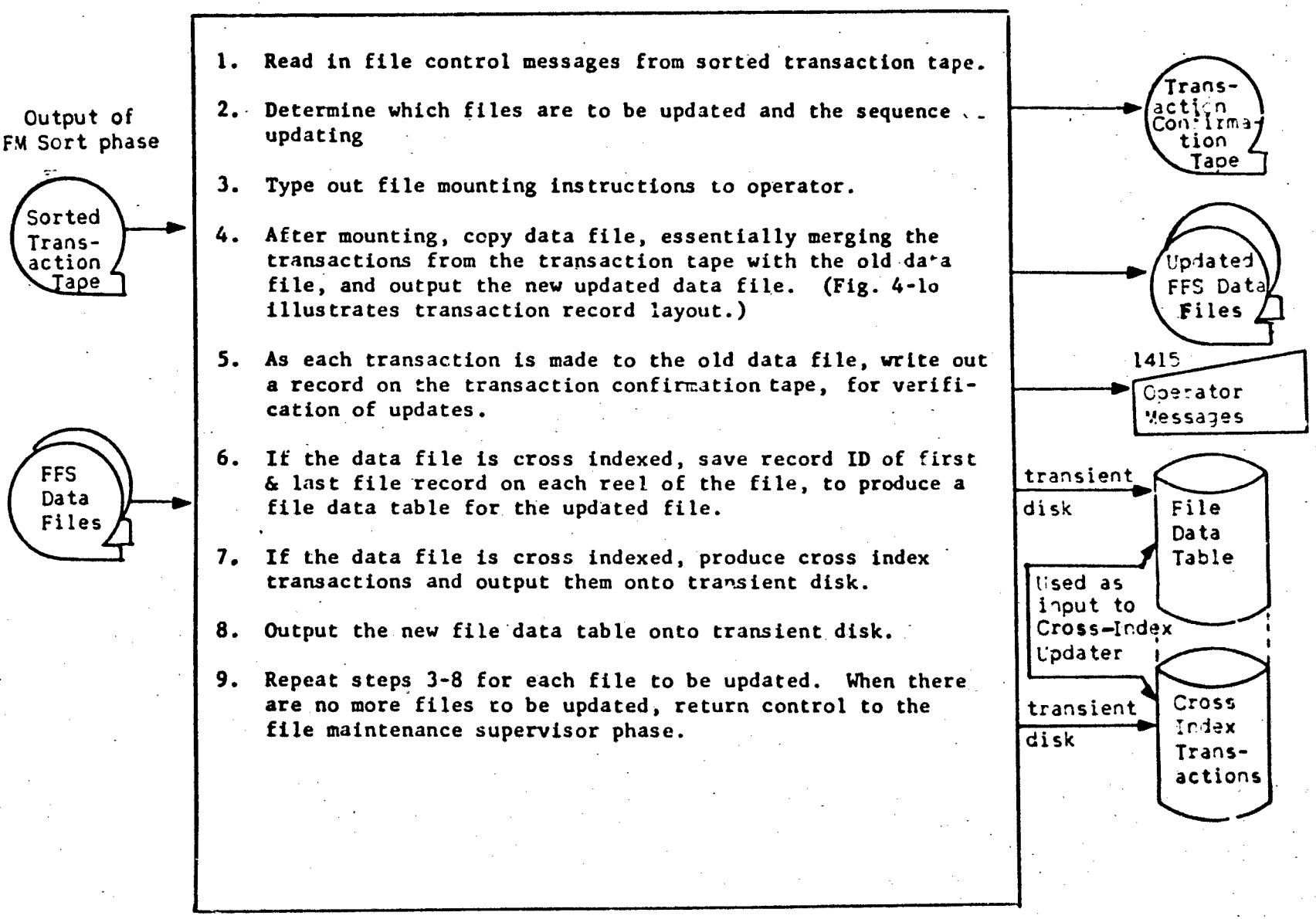


Figure 4-3. General Operation of FM Proper.

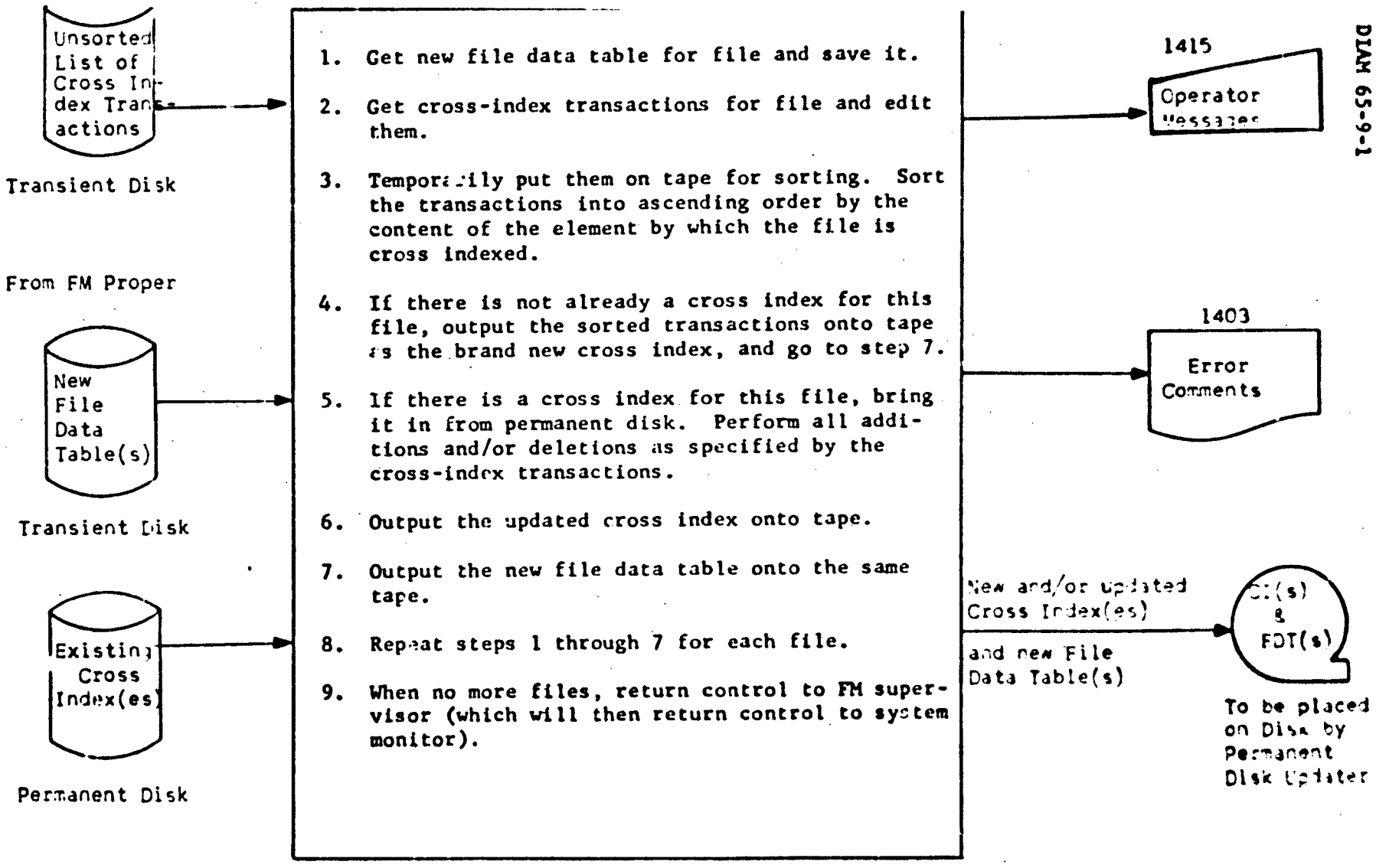


Figure 4-4. General Operation of Cross-Index Updater.

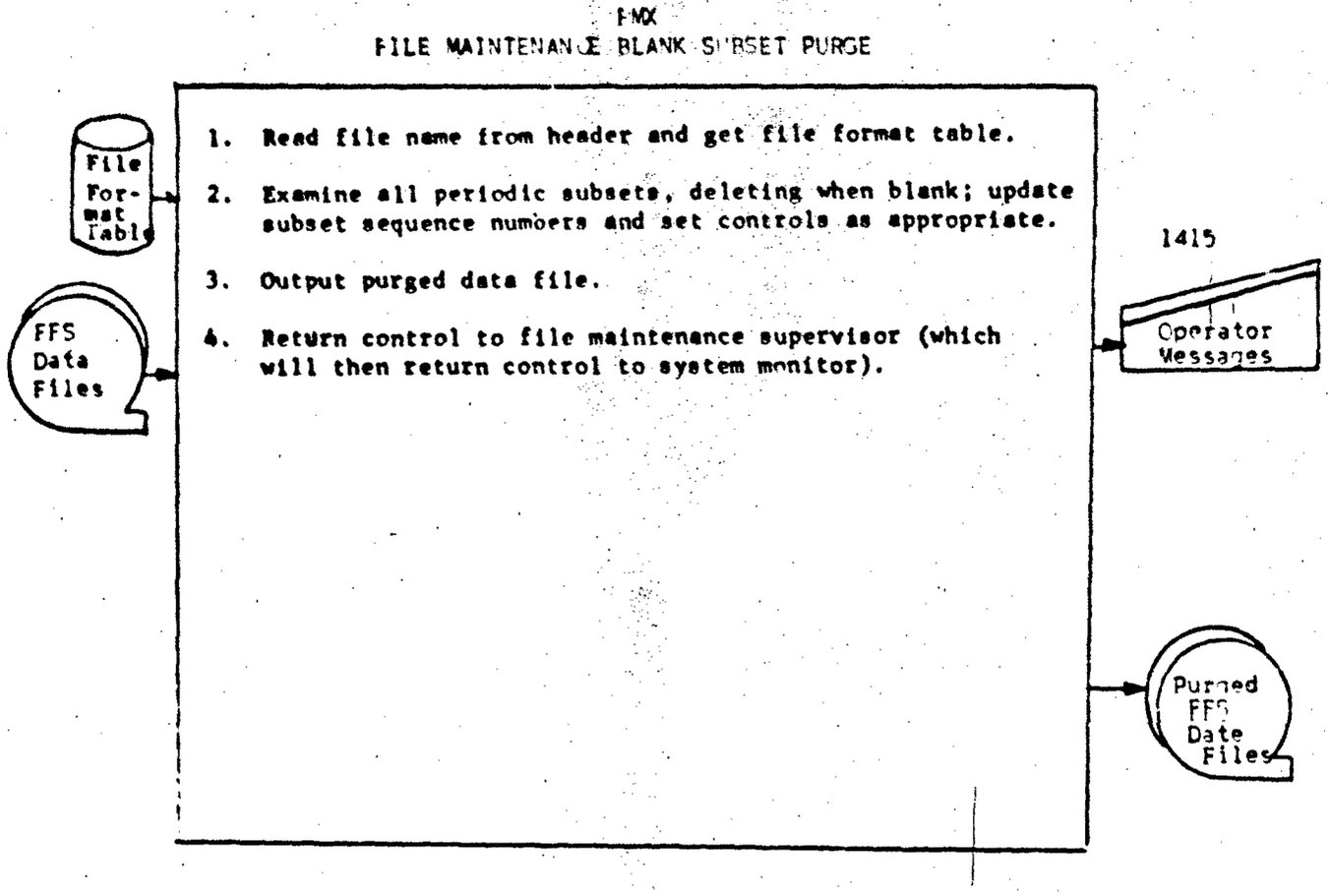
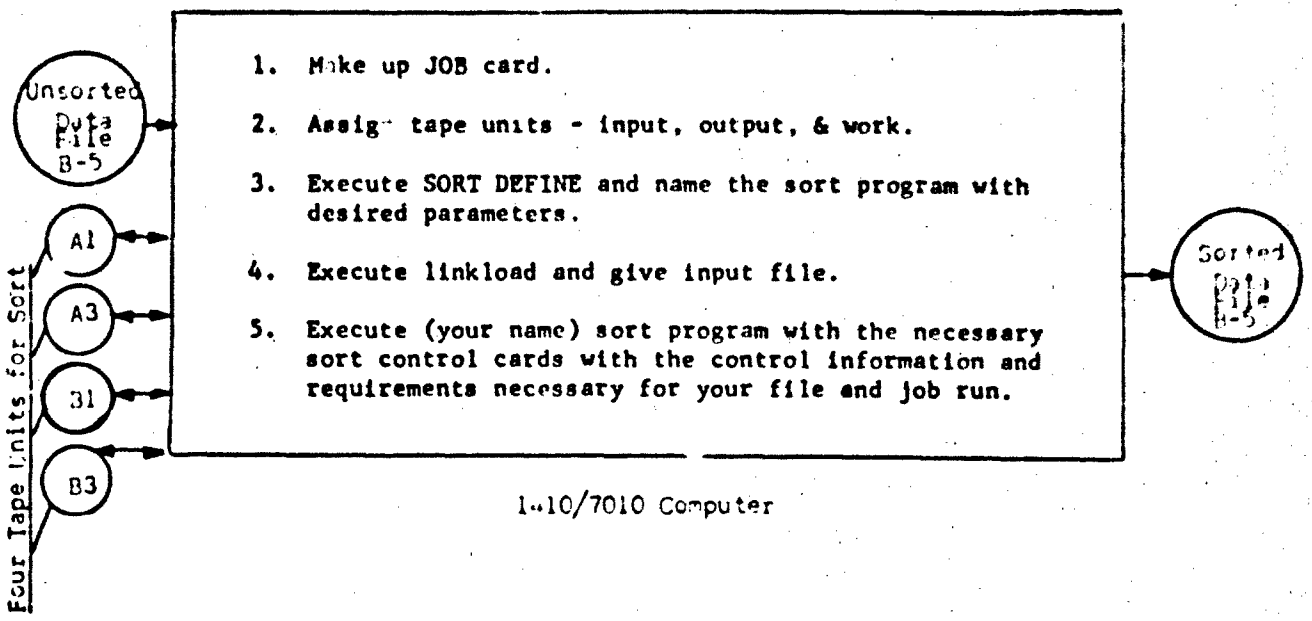


Figure 4-5. General Operation of FMX.

a. Change Record ID - Then resort data file. In Mark II we can make changes to the record ID, or any element within the record ID. Upon completion of a file maintenance run that performs changes on the record ID we will normally have to put the data file back into sort. There is no automatic sort within the FFS programs to perform this sorting job. A sort program called FILESORTER has been generated for this job using the 1410/7010 Operating System Generalized Tape Sorting Program (IBM Systems Reference Library File No. 1410/7010-33 Form (28- 0354). The FILESORTER program will be part of your system operating file (SOF) and executed after every FM run that makes changes to the data file's record ID.



1-10/7010 Computer

Example of Sort Definition Program (FILESORTER) and Workable Sequence of Sort Control Cards:

6	15 16	20 21
MON$$	JOB	TEST SORT FOR MARK II FFS FILES
MON$$	ASGN	MRX,B5
MON$$	ASGN	MRY,A1,A3
MON$$	ASGN	MRZ,B1,B3
MON$$	ASGN	MW2,A2
MON$$	ASGN	MJB,SJ
MON$$	EXEQ	SORTDEFINE
FILESORTER	DSORT	SORT,VARIABLE,MULTI,UNMOD,PCH
	DEFIT	MRX,MRY,MRZ
MON$$	EXEQ	LINKLOAD
	INPUT	MW2
MON$$	EXEQ	FILESORTER
SORTTYPE	SORT	REC-5400,MER-2,OPT-Y,CHK-Y
INPUTFILE	SORT	REC-4,BLK-5404,CHA-4,LOC-4
OUTPUTFILE	SORT	BLK-5404
CNTLFLDS	SORT	NUM-1,LEN-30,1LOC-34,1LEN-30
LABELDES	SORT	TYP-2,ICH-NNNNNN
LABELDES	SORT	TYP-2,OCH-NNNNNN,OFI-CMFLAXXXXX
MON$$	END	

b. Entering Data. Basically there are three forms in which data ients may be entered into data files. Fields, groups, or subsets that to be entered into the file by file maintenance may be:

(1) Placed into the file in the same form in which they were input to file maintenance.

(2) Edited, to insert, modify, or take away certain characters prior to entry into the file.

(3) Converted to a different term or form, or operated upon in any manner, by using a subroutine or table.

c. Transactions Which May Be Performed. The transactions which file itenance can perform are:

Creation of a new data record and the insertion of it into the proper place within the file.

Addition of a periodic subset to a file record.

Addition of data to the variable set of a file record.

Changing the data content of a fixed or periodic field of a file record.

Deletion of the data content of a fixed or periodic field or group. (Actually replaces the data with blanks.)

Deletion of an entire file record.

Deletion of all the data content in a periodic set of a file record.

Deletion of all periodic subsets having blank data content.

Deletion of all data content in a periodic subset of a file record.

Deletion of all data content in the variable set of a file record.

Production of transaction confirmation listing on specified element without affecting data record (NEX operator).

(1) The Formatted File System contains two multi operators - CAD and . The CAD operator will either CREATE (if no file record exists), NGE (if data exists in the file record), or ADD (if the periodic subset s not exist within the file record). The CMA operator will either CHANGE ADD the elements on the basis of their present conditions within the data e. (The use of these operators is subject to the conditions described hin figure 4-8 Summary of Legal Operations that May Be Specified for ments of a File Record.)

(2) By performing multiple transactions (in a separate run or, in some cases, the same run) additional tasks may be accomplished. For example, in order to change the content of the variable set it must be deleted, and then the "new" data (actually the old data with changes) added (during the same run in this case).

d. Input Format Types. Data elements for a file record may be entered into the system in any one of three types of formats. We will now discuss two of these formats which are internal format and external format. Either type of data input is preceded by an input control card (ICC) which provides information concerning the name of the file to be maintained, the form of the input data, the input source number, and other information. The third type of input format is logical file maintenance (LFM) which will be covered in a separate section of this manual.

(1) Internal Format.

(a) For any given file internal input finds its greatest use in updating individual data elements in a file. One standard input card format has been established for this type of file maintenance input. Each internal format input card contains the data to be entered into a single element of a file record, plus identifying (control) information to get it to the correct file record and the correct element within the file record.

(b) The original entry of those file records created locally (internally) by the users of the IDHS 1410 FFS will generally use a constant unchanging format. Maintenance of such file records would normally be accomplished by using internal format input, although external format input might be used under some conditions. Since only one element may be input or updated per internal format (I.F.) card*, several I.F. cards or card images on tape are required per file record. The record ID associated with the element must be present in every I.F. card. In the case of a long record ID, this leaves a limited amount of columns for data content. The detailed specifications of the internal format input cards will be given later in this section.

(c) The user may sometimes resort to the internal format method for the initial creation of his file; although it is usually much faster and more economical to initially create a file from an existing data base by using external format.

(2) External Format.

(a) The data used to update a given FFS data file may be received from a number of different external sources. Each source probably has the data arranged in such a way as to best satisfy the needs

* It should be noted that the internal format card may be in the form of card image on tape.

peculiar to that specific organization. For this and other reasons the format of the data received is likely to vary from one source to another. Also, the formats may periodically or intermittently change due to changing environments of the external sources.

(b) In order to remain flexible in this type of situation file maintenance has been designed to accept external format (E.F.) input data in an extremely wide variety of formats and still be able to enter it into the data file in the format required for the data file. In order to achieve this flexibility the format, location, and identification of each E.F. input data element must be described, as it appears in the input record, to file maintenance. This detailed description of the E.F. input records is accomplished by preceding the external format data with input descriptor cards. These input descriptor cards form a deck called the input descriptor deck (IDD). An IDD must precede each E.F. input file*.

(c) If an FFS data file is being created from an existing data base on cards or tape, the use of an IDD and external format input may well allow the FFS data file to be created from the existing data base without first having to alter either the format or the content of the data base. If this is to be accomplished, the format of the existing data base must be capable of being defined to the system by the IDD, and it must meet certain restrictions which will be discussed later.

4-3. SPECIFICATIONS FOR INTERNAL FORMAT INPUT: Internal format (I.F.) input is used to create or update a single data element of a file record per I.F. input data card. The I.F. input data cards, or card images on tape, for a given file are grouped together into an "I.F. data deck." Each I.F. data deck must be preceded by an input control card (ICC).

Specifications for the Input Control Card (For internal Format)

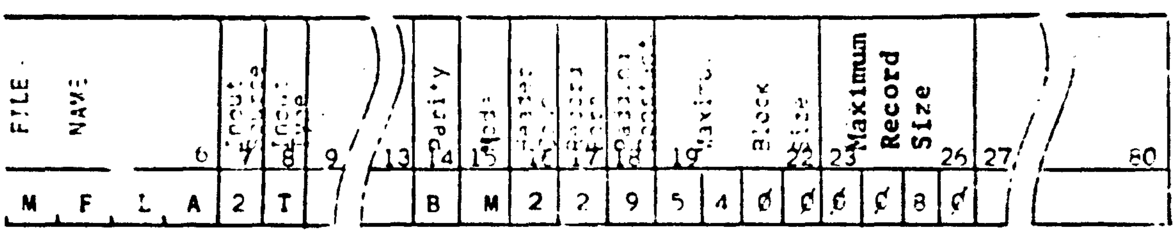


Figure 4-6. ICC for Internal Format

Column 01 = Card identifer △ (11-7-8 Punch).
Indicates that this is an input control card.
Columns 02-06 = Name of file being updated.
Column 07 = Input source must be "2" (internal format).
Column 08 = Input Type. "C" means input data is on CARDS.
"T" means input data is on TAPE.
Columns 09-13 = Blanks

* "Input file" defined on page 4-20.

Columns 14 through 26 are Blank Unless Input Data is on Tape

Column 14 = Parity of tape. "U" for even parity.
"B" for odd parity.

Column 15 = Mode of tape. "L" for load mode (wordmarks).
"M" for move mode (no wordmarks).

Column 16 = Total number of header records (including tape marks associated with the header records). Thus, if a header record and a tape mark precede the data, the number 2 would be entered in column 16.

Column 17 = 1410 IOCS record form (1, 2, 3, or 4).
Form 1 - unblocked, fixed, or variable length.
*Form 2 - blocked, fixed length.
Form 3 - unblocked, fixed, or variable length, with a character count field comprising the first four characters of the record. This field which specifies the number of characters in the physical record (block) is called the "block character count."
**Form 4 - blocked, variable length, with each logical record having a 4-character record count field, and the physical record beginning with a 4-character block character count field. The record character count field specifies the number of characters in the logical record.

Column 18 = Padding constant to be used (when required) to fill out the block, if form 2 records are specified. Nines (9's) are generally used for padding. The four special characters ✓ ≠ ≠ * cannot be used for padding.

Columns 19-22 = Maximum block size, maximum number of characters that may be encountered in a block, if form 2 or form 4 records specified. Maximum allowed is 5400, which may be used if the actual maximum block size is unknown.

Columns 23-26 = Maximum record size, normally 80 (81 if blocked), to be encountered, unless form 4 records are specified; in which case this entry specifies the relative LOP of the record character count field. (For this purpose, the first position of a form 4 record is considered 1, not 0.)

Columns 27-80 = Blanks

Specifications for the Internal Format Input Data Card

Figure 4-7 gives the specifications for the internal format input data card.

* NOTE: Form 2 records must have each logical record terminated with a record mark.

** NOTE: Form 4 records must have each logical record terminated with a record mark.

I.F. INPUT DATA CARD

SEQ NO	ELEMENT NAME	OP CODE	PSSQn	VARIABLE FIELD	
1 2	3 7	8 10	11 13	14 4344	7380
ØØ	GEREF	CHG	ØØ2	AMERICANBOSTONØØØØØØ64122614332‡4132 1	W‡
ØØ	FLITE	DEL	ØØ2	AMERICANALBUQUERQUEØ64122523122‡	
ØØ	PSET2	ADD	Ø1	UNITEDØØFRIENDSHIPØØ6503141385 X ‡ØØ1DE	Ø‡

Figure 4-7. Internal Format Input Data Card

Cols. 01-02 = Sequence number, if desired. Not used by the FFS at all, but may be used externally.
Cols. 03-07 = Name of data element (field/group/subset).
Cols. 08-10 = Operation code. (CAD, CMA, NEX, ADD, CHG, CRE, or DEL.) See Figure 4-8 for detailed explanation of Op Codes.
Cols. 11-13 = Periodic Subset Sequence, if element to be maintained is periodic. If element is fixed or variable, leave these columns blank. If the Op Code is ADD or CRE the next sequential PSSQn will be assigned to periodic data having blanks in these columns. If Op Codes CAD or CMA are used, the next sequential PSSQn may be assigned if they are executed as an ADD or CRE Op Code. WARNING: the PSSQn number must be furnished if they are to execute as CHG.
Cols. 14-XX = Exact Record ID of file record of which the data element being maintained is a part. Since the Record ID Size may be from 1 to 30 characters in length, this field will not end earlier than column 14, and not later than column 43.
Col. XX+1 = Record mark ‡. (Ø-2-8 punch.) This will be the last entry on the card if Op Code is DEL.
Col. XX+2-YY* = Data associated with element being maintained. Unless Op Code is DEL; then leave blank.

*NOTE: If column YY exceeds column 79, the use of a continuation card is required. To indicate that data is continued on the next card, a > (6-8 punch) is placed in column 80. Columns 03 through "XX"+1 of the continuation card are duplicated from the preceding card. The data is then continued from column "XX"+2.

Col. YY+1 = Record mark ‡ (Ø-2-8 punch).

SPECIAL NOTE: Be sure to see additional comments concerning this card on the next page.

Entering Internal Format Input Data via Magnetic Tape

Card images on tape may be used for I.F. input via a tape unit instead of punched cards via the standard input unit (SIU). To accomplish this the ICC card is read from the SIU, the FM program then obtains all the necessary control information from the data on the ICC card and reads the I.F. input data card images on tape.

4-4. ADDITIONAL COMMENTS CONCERNING THE DATA ENTRY IN THE I.F. INPUT DATA CARD:

a. Numeric Fields. Leading zeros may be omitted for those elements which have been described as numeric in the file format table for the data file being updated; e.g., if a 5-character numeric data field is to be changed to 00037, then the data entry on the internal format card could be entered as 37 ǂ. The file maintenance program would automatically insert the 3 leading zeros. CAUTION.....it is not safe to assume that the data file field has been defined as numeric simply because the input data is numeric in content! When in doubt pad out the data entry to its proper data file length by inserting the leading zeros. If zero substitution for blank input data is not desired, the field should be defined in the FFT as Alpha.

b. Alphameric Fields. An Alphameric data entry may always be terminated (by the ǂ) immediately following the last valid character. The file maintenance program will left justify the entry and add any additional trailing blanks necessary to pad out the field/group to its proper file length.

c. Maximum Data Length.

(1) Ordinarily the size of an input data element cannot exceed the size of the FFS data file element for which it is destined. The exception to this rule is a data entry which undergoes a subroutine conversion before being inserted in the data file. Whenever the input data appears to be too long, and the person making the entry is in doubt as to the validity of the size of the entry, he should consult the file designer about the legitimacy of the input data entry. File maintenance will not attempt to put any input data entry that has an illegitimate size (i.e., too many characters) into the FFS data file. An error message identifying the name of the element in error, and the maximum size allowable is printed out by the IP phase of FM, in such an instance.

(2) When the I.F. data card is used to enter data for the variable set, continuation cards may be used ONLY until the size of the data entr reaches a maximum of 910 characters in length. If the desired entry is longer than 910 characters, it must be entered in two or more "units." Each unit will consist of the initial I.F. data card for that unit followed by continuations until the sum size of the data entry for the unit approaches or reaches 910 characters. The last card of the unit must not contain a continuation symbol. The data to appear first in the variable set must be entered in the first unit; the last data for variable set,

entered in the last unit. In other words, all of the units must appear "back to back" and in the desired sequence.

4-5. SPECIFICATIONS FOR THE IPPAKEND CARD FOR INTERNAL FORMAT:

a. The IPPAKEND card is not required, but is desirable with most I.F. input jobs. Its only function when used with internal format input is to cause the input processor phase of FM to perform an "error computation and statistical error printout" routine at the end of an I.F. input job. If not used with an I.F. input job, this routine is simply not performed for that job.

b. The IPPAKEND card can be optionally used with E.F. input when the input is on cards. However, the above-mentioned error computation and statistical error printout routine is performed whether the card is in the deck or not. At the present time, if the IPPAKEND card is used with E.F. input, when the E.F. input data is on tape, the error computation and statistical error printout routine is performed twice. The specifications for the IPPAKEND card are simply to:

Punch IPPAKEND into columns 1-8.

Leave columns 9-80 blank.

The IPPAKEND card, if used, should appear immediately after the last I.F. data card.

4-6. OPERATION CODES: The seven operation codes that are used to specify the disposition of an input data element are:

Create (CRE)

Add (ADD)

Change (CHG)

Delete (DEL)

Create, Change or Add (CAD)

Change or Add (CMA)

No Change (NEX)

Their use and the resulting actions are illustrated in figure 4-8 (two pages). The use of these operation codes and the actions resulting upon the various FFS data file elements are the same for both internal format and external format.

FFT Entry Type	ELEMENT TYPE	Use Only if File Record Having Record ID Being Maintained Does Not Exist	Use Only if File Record Having Record ID Being Maintained Does Exist					Record ID May or May Not Exist
		CREATE (15)	ADD	CHANGE	DELETE	CMA	NEX (17)	CAD
1	Record Control (or Record ID) Group and or Field	✓ (8)	NO	✓ (18)	✓ (1)	✓	✓	✓
2	(Non-Control) Fixed Group or Field	✓ (9)	✓ (2)	✓	✓ (3)	✓	✓	✓
3	Periodic Field or Group	✓ (10)	✓ (13)	✓ (5)	✓ (3) (5)	✓	✓	✓
4	Variable Set	✓ (11)	✓ (7)	NO (14)	✓	NO (14)	NO	NO (14)
5	Periodic Subset	✓ (10)	✓ (12)	✓ (5)	✓ (5) (16)	✓	✓	✓
6	Periodic Set Control Field	NO	NO	NO	✓ (4)	NO	NO	NO
7	Variable Set Control Field (6)	NO	NO	NO	✓	NO	NO	NO

Figure 4-8. Summary of legal Operations that may be Specified for Elements of a File Record

(0) ✓ = Yes you may use this Op Code for element specified.

Legend to Figure 4-8 on next page.

Legend to Figure 4-8.

(1) This will delete the entire record.
(2) This doesn't actually add an element, it becomes a CHANGE which changes the contents of the element to new data.
(3) This doesn't actually delete an element, it becomes a CHANGE which changes the element to blanks.
(4) This actually deletes the entire periodic set specified, but the periodic set control field is retained, as blanks.
(5) The periodic subset sequence number (ID) must be provided.
(6) FM proper phase automatically creates, maintains, or deletes the variable set control field as required.
(7) If a variable set already exists in the file record being maintained, the variable data being added trails the existing data.
(8) A record having the record ID (control field/group), with blanks in the rest of the fixed set will be created.
(9) A record having the record ID (control field/group), as well as the fixed field(s)/group(s) specified, will be created.
(10) A record having the record ID (control field/group), with blanks in the rest of the fixed set (except the PSCTn field), will be created in addition to the subset.
(11) A record having the record ID (control field/group), with blanks in the rest of the fixed set (except the VSET control field), will be created in addition to variable set.
(12) This will add a new subset, all field(s)/group(s) of which contain the supplied information.
(13) This will add an entire new subset, but data will only be contained in the field(s)/group(s) specified. The other field(s)/group(s) will be blank.
(14) In order to change a variable set, it must first be deleted, then added. This may be done on the same or different runs.
(15) After the first CREATE is performed for a given file record, other CREATES appearing in the same run and applying to that same file record, have the effect of a CHANGE for fixed fields; or an ADD for either periodic elements or the variable set.
(16) In order to delete the <u>entire</u> subset (including the PSSQn), all of the elements of the periodic set (staring with the PSSQn) must have been grouped together by a GROUP card when the FFT was structured. (To put it another way, a type ID of 5 must exist for the subset in LR 2 of the FFT.)
(17) The FFS program handles the NEX opcode in the same way as CHANGE - what is legal for the CHG of code is legal for NEX.
(18) This is legal in 1410 FFS. Every time the record ID is changed an advisory message is printed, on 1403, that the data file may need to be re-sorted. The record ID is also printed.

4-7. DESIGN OF AN INPUT FILE FOR EXTERNAL FORMAT:

a. General.

(1) Once the preliminary design, data content, and use of an FFS data file have been decided upon, the FFS user will probably next structure the FFT for the file using file generation. The next task the user faces is to produce the FFS data file; i.e., get the appropriate data into the file so that it may be used.

(2) The starting point for building a 1410 FFS data file is an input file of data in the form of either punched cards and/or magnetic tape whose format and content can be defined to the Formatted File System.

b. Creation of an External Format Input File.

(1) For some users the input file of data just described does not exist and must be created. This implies that card formats must be designed, appropriate input forms drawn up and filled out, and cards key punched from them to form the input file. When designing card formats for the input file the user may use the "Multiple Card Layout Form" (IBM Form No. X74-4823) as a basic design tool. (Also the "Card Design Aid" - IBM Form No. X24-6214.) It is also strongly recommended that the user be familiar with the rules and suggestions contained in the IBM Manual "Form and Card Design" (IBM Form No. C20-8078).

(2) After completing the card design for the input file, the user must design the input forms used to record input data for key punching. The design and content of these forms is at the discretion of the user; however, he should keep the following points in mind.

(a) The sequence of entries on the forms should follow some logical pattern. This logical sequence will be determined partly by the arrangement of the information on the original source documents used by the person who fills out the input for-

(b) The input forms should be set up for ease in key punching. The ideal form, of course, would have the form entries arranged in the same sequence as they appear on the card.

In many cases, the above two points can never be fully realized because of conflicts between the arrangement of data on the original source documents and the arrangement of fields on the input cards. The user should attempt to strike some happy medium between the two.

(c) If some data elements must enter the system via the internal format method (due to restrictions to be covered later) the format for these I.F. input cards should be included on the external format input forms along with the other data elements. Any control information that remains constant should be preprinted (e.g.: field/group name, record marks, continuation symbols, etc.).

addition to the above points, refer to the IBM Manual "Form and Card ign" (IBM Form No. C20-8078) for a more complete discussion of the topic forms design.

(3) In the above situation, since the user creates the input file data with full knowledge of its purpose, he will make it compatible with FFS data files. In fact, under these circumstances, it would not be prising if the input file of data very closely resembled the FFS data e to be produced.

c. Using Existing Files of Data as E.F. Input Files for FFS Data File Creation.

(1) On the other hand, some users facing the task of producing an FFS a file may already have existing (probably Punched Card Accounting Machine AM] oriented) files of data on cards and/or tape, whose format and content d only be defined to the FFS by an IDD in order for FM to produce the FFS data e. (The existing data files are used to form the E.F. input file.)

(2) The user should be familiar with the design of input files external format whether he has to create his input file, or whether already exists in the form of PCAM oriented files of data. This is ecially important to the user who desires to produce an FFS data file using his existing files of data as the E.F. input file to file main- ance. He must be able to determine whether his existing files of data be sufficiently well described by an IDD to be handled by FM in its sent form. It may be that the particular format of his existing files data precludes the direct acceptance of parts of the information by the matted File System. The user may have to either reformat these por- ns by using punched card processing machines or resort to the internal mat input method for some elements in order to conform to the FFS uirements for the E.F. input file.

(3) The next portion of Section Four contains the definitions terms used in the rest of this section, followed by the design con- erations and requirements for any external format input file that is to accepted by file maintenance. It also provides the ground rules for igning an E.F. input file. The emphasis so far has been on using E.F. ut to initially produce FFS data files. Do not conclude from this that E.F. input is not particularly suited for routine maintenance of sting FFS data files. This use of the E.F. input method will be covered er in this section.

. GENERAL INPUT FILE REQUIREMENTS AND DEFINITION OF TERMS: The follow- , definitions will provide the user with some of the terms used in this tion as well as some of the overall design requirements for an input e.

a. Input File. A card or tape file which contains all or a portion the input data needed to create or update a given data file. Each E.F. ut file must be preceded by an IDD.

b. Input Record. A single card (or logical tape record) in an input file

c. Input Record Type. An input record whose format (arrangement and location of data fields) can be identified by the presence of a unique code carried in the input record itself. For example, if there are three different types of record formats employed in the input file, there are three input record types, each type identified by an "input record type code."

d. Input Record Type Code. The code used to distinguish one input record type from another.

(1) If the input file contains multiple input record types, a unique code must be present in each input record.

> Exception. If the input file contains only one input record type, no input record type code field need be present in the input records.

(2) The input record type code field size may vary from 1 to 7 positions between files but must remain constant within the same input file.

(3) The input record type code field must appear in the same position(s) in every input record.

(4) The input record type code must be the only data in its field. In other words, the input record type code cannot be zone punches placed over a numeric data field.

(5) The makeup of input record type codes may be alphabetic, numeric, or alphameric and may vary within the same input file.

e. Input Group. All of those input records containing information be extracted for the purposes of creating or updating a single (the same) file record.

f. Input Group Control Field. Either an artificial control field (such as an arbitrarily assigned serial number) or an actual data field (or fields) upon which the input file will be sorted or manually arranged, prior to entering the input file into the system, such that all input records belonging to the same input group (i.e., pertaining to the same file record) will be grouped together.

(1) All input records must contain an input group control field.

(2) The field must appear in the same location in every input record in a given input file.

(3) Each input group within an input file must have a unique input group control field. (The input records within the input group will, of course, have the same input group control field.)

(4) The input group control field must be a field, i.e., all the characters comprising the input group control field must be adjacent in the input record. In other words, if the user intends to use a number of data fields as the input group control field, these fields must be adjacent to each other so that the system can treat them as a single field.

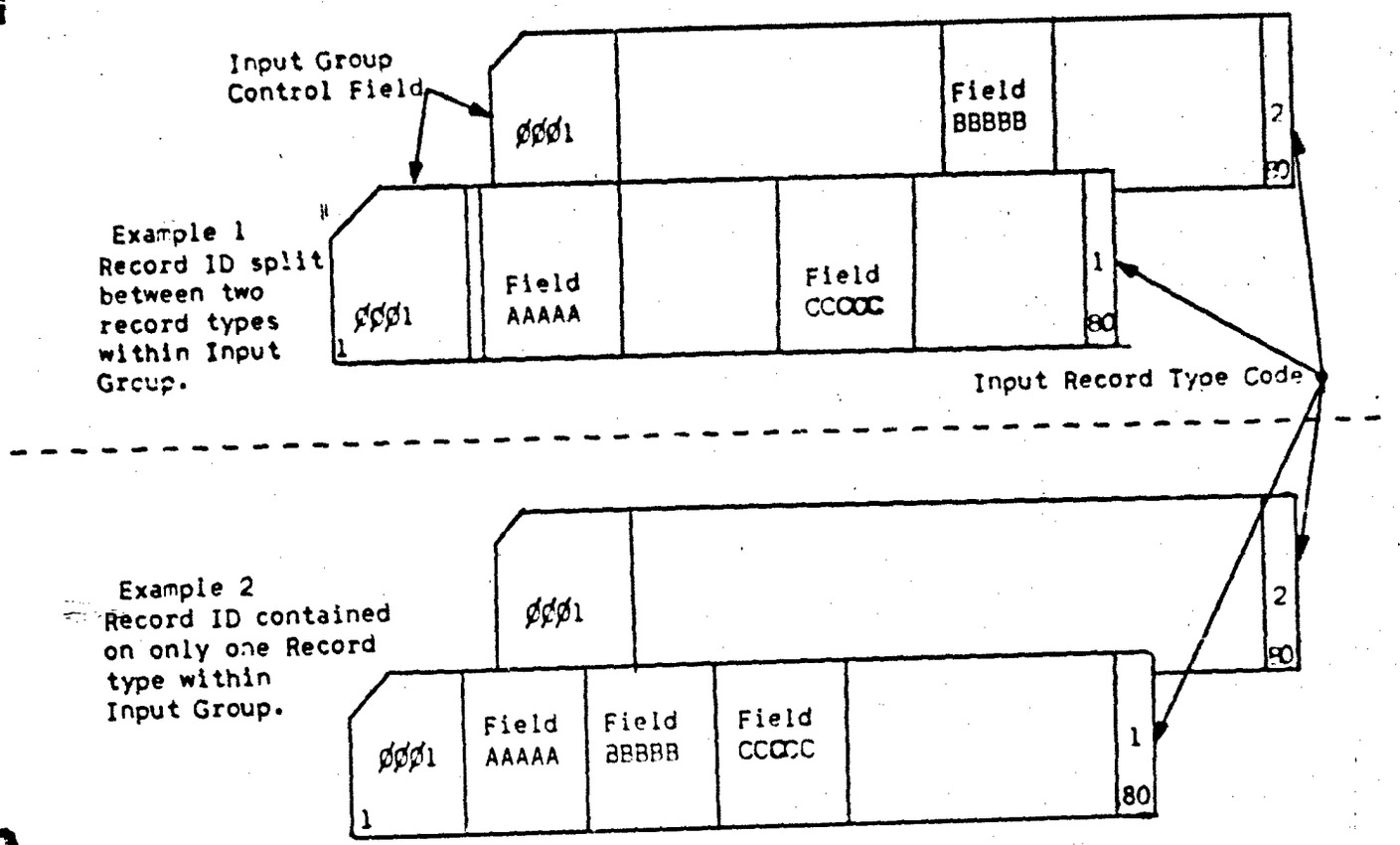
(5) If the input group control field contains an artificial number (such as an arbitrarily assigned serial number used for the singular purpose of distinguishing one input group from another), then the field size will be determined by the maximum number of input groups to be processed against the data file in one FM run. For example, if a maximum of 10,000 input groups are to be processed against a given data file during a file maintenance run then the input group control field need only be 4 characters in length. The input group control numbers would commence with 0000 and end with 9999.

NOTE: It is very important to note that the data file records will not necessarily be ordered (sequentially arranged) by the contents of the input group control field unless it is one and the same as the field(s) comprising the record ID. It is simply a device for grouping together (prior to processing) all of the input records affecting single data file records.

g. Record ID (Record Control Group). The record ID is the field or fields whose data content makes up the unique identity of each data file record. It is sometimes referred to as the "control group" or "record control group."

(1) The field or fields to be extracted from the input records forming an input group and assembled into the record ID can be spread over several DIFFERENT input record types within the input group. However, all the fields that comprise the record ID must be present within each input group, present in the input file (and if tape input, not split between tape reels).

For example: Assume an input file with 2 input record types (coded as 1 and 2) per input group, and containing fields AAAAA, BBBBB, and CCCCC (which comprise the record ID). Any one of the following formats is permissible for the input file.



(2) In example 2, it is obvious that the 2 type record need not be present in the input group since the record ID can be extracted in its entirety from the 1 type record. Thus, if the data ordinarily carried on the 2 type record is not required or unknown for a particular file record, the 2 type record can be legitimately omitted from the input group.

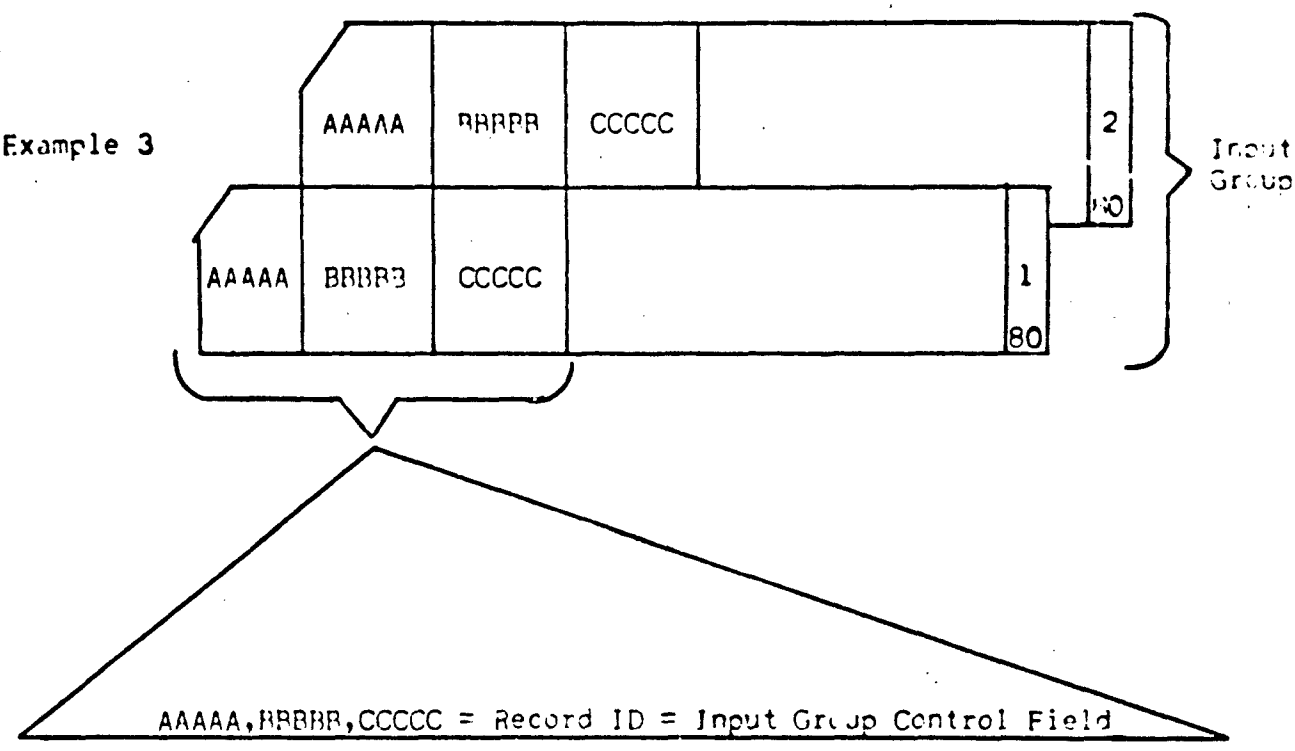
(3) As a general rule, record ID fields should be placed on input record types carrying fixed data rather than periodic data, unless the record ID is used as an input group control field. This use is described in the following example.

(4) When initially designing an input file, the user should seriously consider using the record ID fields as an input group control field. The very nature of a record ID insures its uniqueness. If the user does design his record ID to serve this dual purpose, he must remember that each field that comprises the record ID must be present in every input record and in

the same location, and, in addition, the fields must be adjacent* to each other.

For example:

Example 3



(5) The user should be aware that the above approach could be spacewise; i.e., if in example 3, fields AAAAA, BBBBB, and CCCCC led 30 characters, then obviously 30 columns of every input record d immediately be lost, and only 50 columns would be available for fields and record type code (using cards as input records). In a case, it might be better for the user to reserve a smaller

E: If an existing file of data (to be used as an E.F. input file) have the record ID fields in every input record but not in cent positions, the card file should be reproduced with the record ields placed adjacent to each other, utilizing punched card proc- ng equipment.

field for the input group control field (using a serial numbering technique) and spread his record ID field over several record types. (See example 1.) When input records are from tape rather than cards, it is not much of a disadvantage to use up to 30 characters of each input record as record ID/input group control field, since the tape input records may be longer than 80 characters.

4-9. PREFERRED LOCATION OF CONTROL FIELDS ON INPUT RECORDS: If possible, the user should arrange his input record format so that all control fields (input group control field, record type code, record ID) will be the first (left most) fields on the input record. Data fields should immediately follow these control fields. This is not a requirement, but a recommendation which will result in a more efficient, less error prone, and faster key punching operation.

4-10. SPECIFIC INPUT FILE REQUIREMENTS (EXTERNAL FORMAT): This subsection explains the principles involved in designing input record card formats which will contain:

Fixed Fields/Groups.

Periodic Fields/Groups.

Periodic Subsets.

Variable Set Information.

a. Fixed Fields. Fixed fields may be present on any input record type regardless of the nature of the other information on the input record. However, it is preferable to design fixed fields so that they will not appear on the same record type that carries periodic or variable information. (The one exception being the case of using the record ID as an input group control field, in which case it is on every card of the input group.)

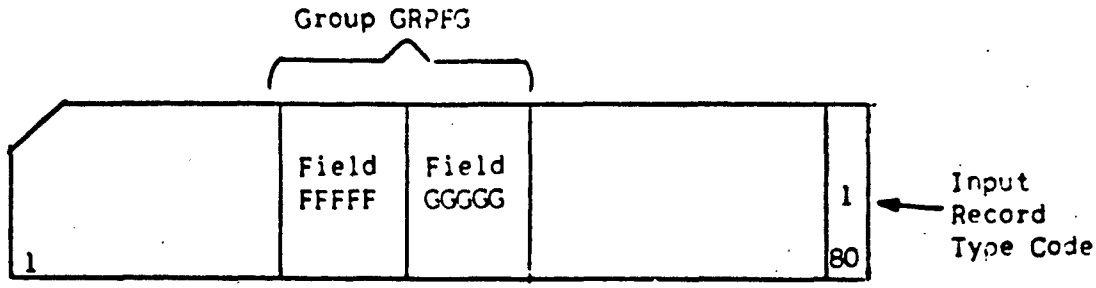
b. Splitting Fixed Fields. If the size of a data field exceeds the space available on 1 external format input record, the field can be split over two or more different input record types, if the data field represented is raised to a group level within the FFT for the FFS data file, and each field of the group entering via a separate record type, is defined as a field in that group. If it is not desired to split fields in this manner (for example, if the input field must be altered by an input conversion subroutine that requires the entire field at once), the internal format method may be used to enter or update such fields, since by using continuation cards a very large field may be handled.

c. Fixed Groups.

(1) When a number of fields appear in the file's FFT as a group, it would be desirable for the user to place these fields on the same input record type and arranged in the same order as they appear in the file record.

For example: Fields FFFFF and GGGGG are considered a group (GRPFG) in the FFT. The preferred input format would be:

ample 4



(2) (Design fields FFFFF and GGGGG adjacent to each other on ıe same input record type and in the same order as they appear in the ata file.)

(3) The desirability of this approach stems from the fact that ıe processing of groups through FM is faster than the processing of the ıdividual fields comprising groups.

)TE: If any one of the fields within the group must be processed by a ıbroutine prior to being inserted in the data file, the above recommen- ıtion would be disregarded, since the fields have to be handled indi- idually in this case.

SPECIAL NOTE: No field which forms part of the record ID can be processed by an input conversion subroutine.

d. Splitting Fixed Groups.

(1) If a number of fields are grouped (in the FFT), AND

(2) the fields must undergo an input subroutine conversion ı a group prior to being inserted in the data file, AND

(3) the fields, because of their sum size, cannot be contained n a single input record, then they will have to enter the system (as a roup) via the internal format method.

hen only conditions (1) and (3) are present, the fields may be defined nd input (via E.F.) individually.

hen only conditions (1) and (2) are present, the fields may be input as a roup (which they are defined as in the FFT).

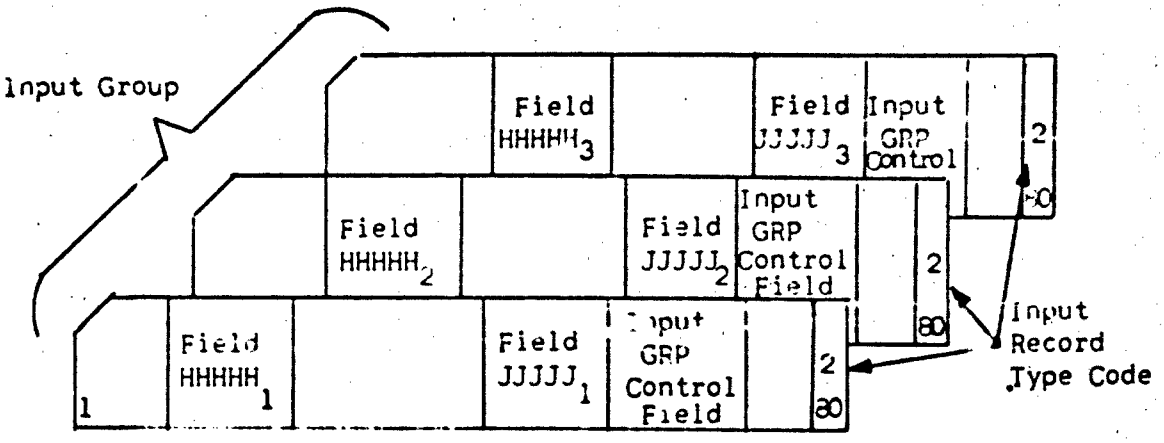
e. Periodic Fields/Groups. Periodic fields/groups may be present on ny input record type, even if fixed fields/groups appear on the same input ecord. All periodic information on an input record type must:

(1) Belong to the same subset, and

(2) belong to the same set.

Condition (1), of course, would guarantee condition (2). (The one exception to (1) above will be noted later under Multiple Subset Entries.)

For example: Assume fields HHHHH and JJJJJ to be periodic fields in periodic set 1 of the FFS data file.



Example 5

Fields HHHHH and JJJJJ appear on all 2 type Records

Assuming a CREATE opcode:

$HHHHH_1$ and $JJJJJ_1$ become fields in the 1st subset.

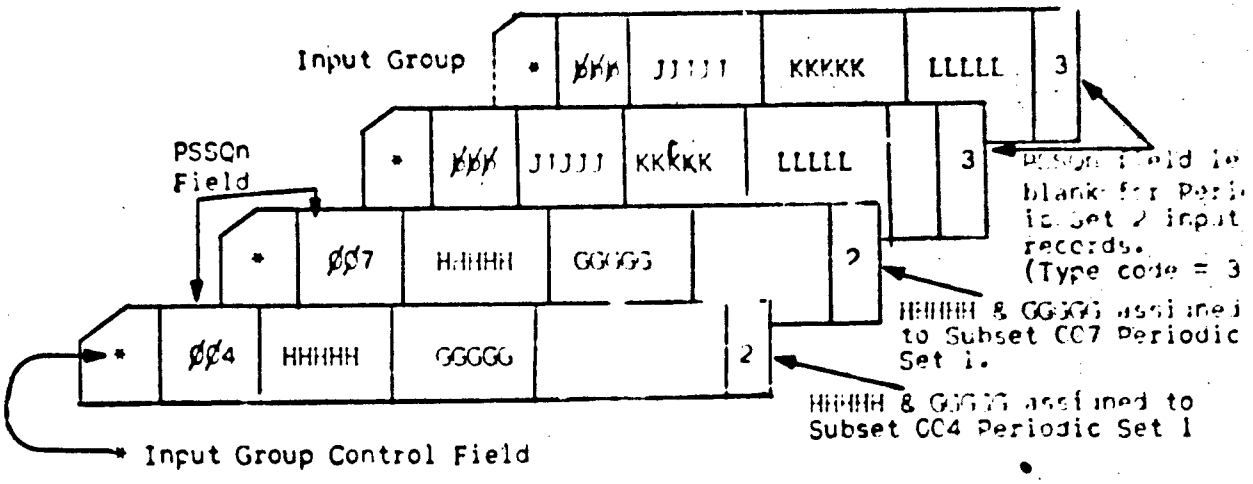
$HHHHH_2$ and $JJJJJ_2$ become fields in the 2nd subset.

$HHHHH_3$ and $JJJJJ_3$ become fields in the 3rd subset.

f. Periodic Subset Sequence Number (PSSQn).

(1) If periodic fields/groups are to be assigned to a specific subset, a 3-character field must be provided on the input record for entering this information.

(2) This field must be in the same location on every input record type carrying periodic information. In other words, once the decision has been made to assign specific periodic set control numbers to the periodic field/groups in one periodic set, the PSSQn field must appear on the input record types carrying periodic information for other periodic sets. (Although the latter does not have to be assigned specific PSSQn's - See example 6 below.)



Example 6

(3) If the PSSQn field is left blank, it is said to be unassigned and file maintenance will automatically assign the next available PSSQ number available in the periodic set involved. In Example 6, for instance, the first JJJJJ, KKKKK, LLLLL entries would be assigned 001 and the second JJJJJ, KKKKK, LLLLL entries 002 during a data file creation. During an updating process the next 2 available numbers in periodic set 2 would be assigned; e.g., 009 and 010 if the last periodic subset in periodic set 2 of the specified data file record were 008.

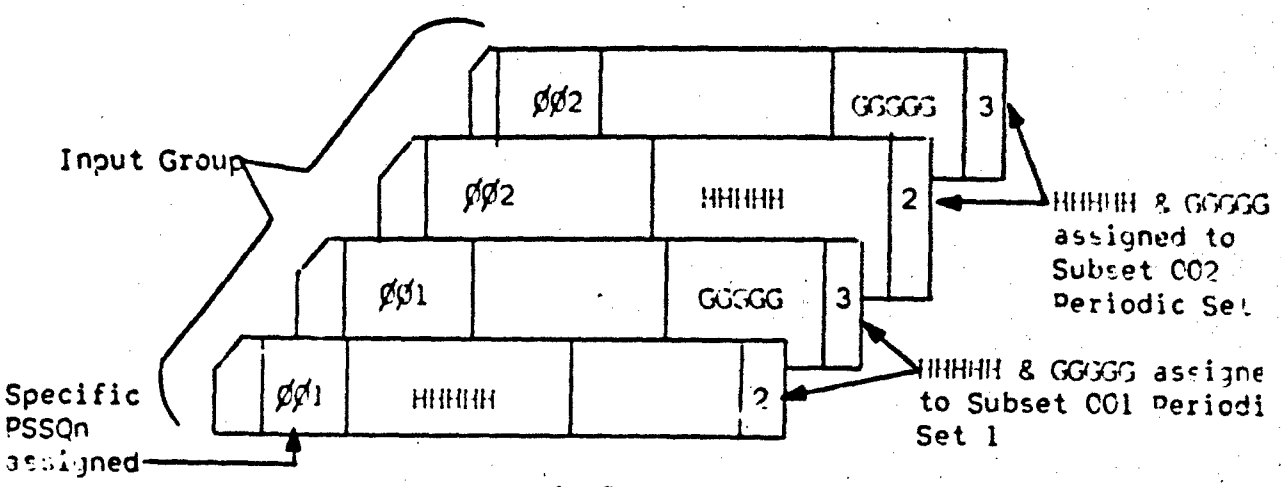
(4) The order in which unassigned periodic information records are input to file maintenance determines the order in which the periodic information is placed in the data file (assigned PSSQn's), unless pseudo numbers are used.

(5) If all the periodic fields/groups for a periodic subset exceed the space available on one input record, they can be spread over 2 or more DIFFERENT input record types. In this case the user must:

(a) Always provide a PSSQn field in every periodic input record, AND

(b) always assign a specific or pseudo PSSQn for the entries in the periodic set which is split. (Cannot leave the PSSQn field blank.)

For example: Assume periodic set 1 fields HHHHH and GGGGG together exceed one card.



Example 7

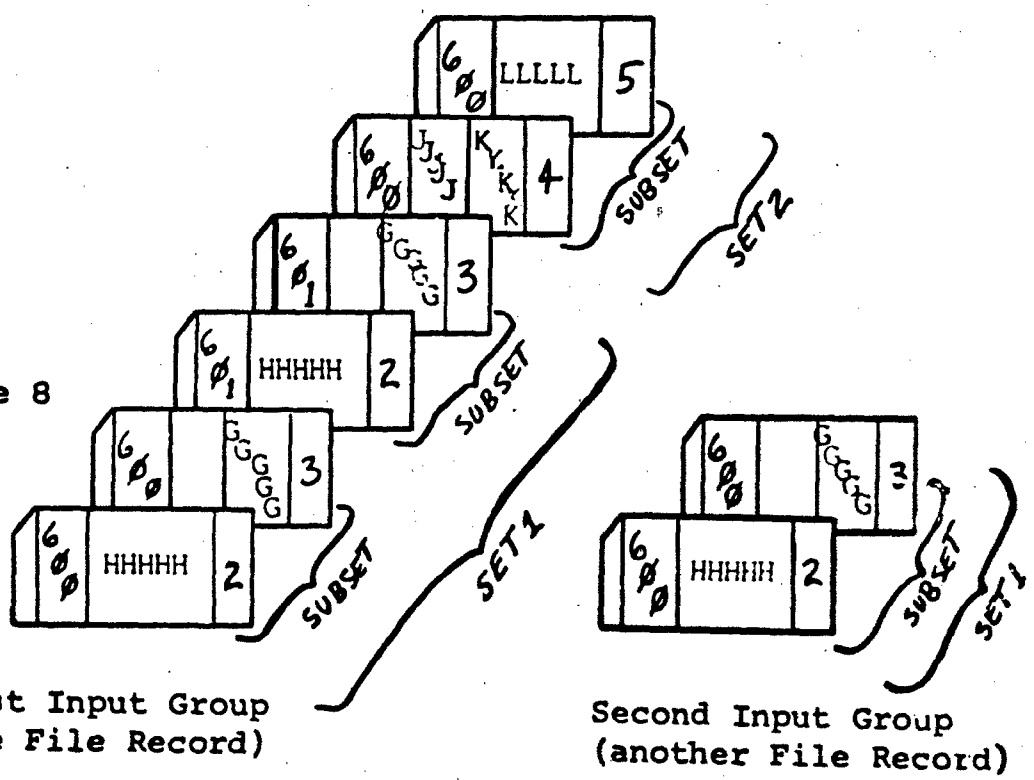
(6) In the above example, if the user wishes file maintenance to assign the next available PSSQn's in the periodic set, he must insert a pseudo number in the PSSQn field. The pseudo number may be any number ranging from 600 to 999. As with specific PSSQn's the pseudo number must be:

(a) Unique for each periodic subset within a periodic set,

(b) the same for all elements destined for the same periodic subset.

The pseudo number does not have to be unique between different periodic sets or between different input groups. See example 8 below.

Example 8



First Input Group (one File Record)

Second Input Group (another File Record)

g. Periodic Fields/Groups - Split. The same rules stated in splitting fixed fields and groups apply to periodic data entries, with the additional implications being those already mentioned under periodic subset sequence number.

h. Subset Entries - Multiple and Single.

(1) Periodic fields for more than one subset can be contained on the same input record type if:

(a) All the fields comprising the subset are present on the input record, AND

(b) none of the fields have to undergo an input subroutine conversion, AND

(c) the FFT for the data file contains a periodic subset entry (type ID of 5 in LR #2).

For example: Assume that periodic fields HHHHH & GGGGG together total 5 characters in length and that neither field has to be converted by a subroutine prior to insertion in the FFS data file.

(2) First - The FFT must contain a periodic subset entry which includes fields HHHHH and GGGGG and the PSSQn field. (It is similar to an FFT group entry, but includes the PSSQn file field name.)

The GROUP card which caused the periodic subset entry would start off as follows:

BOTHØ GROUP PSSQ1, HHHHH, GGGGG

where; BOTHØ is the subset name,

PSSQ1 is the PSSQn field name for periodic set 1

HHHHH and GGGGG are the names of the periodic fields which comprise a subset in periodic set 1.

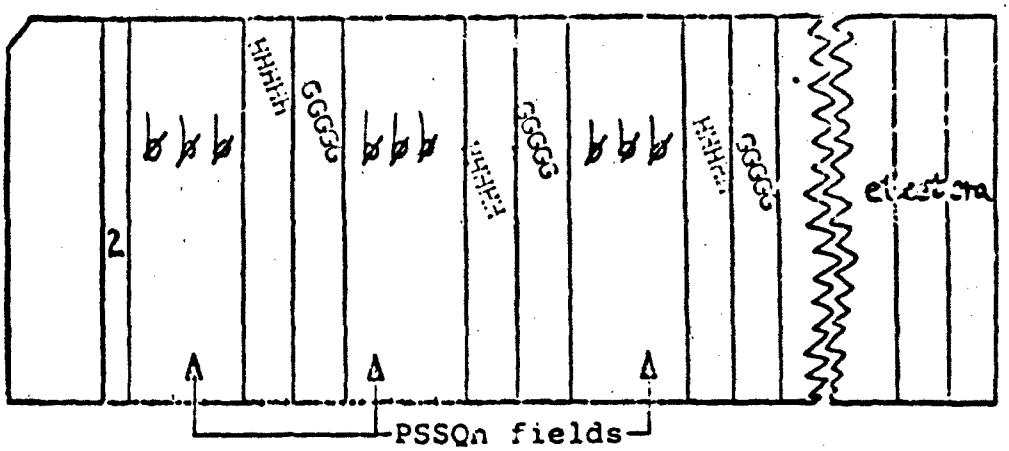
(3) Second - Multiple periodic subset entries can then be spread across one input record type in the following manner:

All of the fields for each subset must be adjacent and in the same order as they appear in the file record.

Each group of fields for one subset must be immediately preceded by a 3-character PSSQn field.

For example:

Example 9



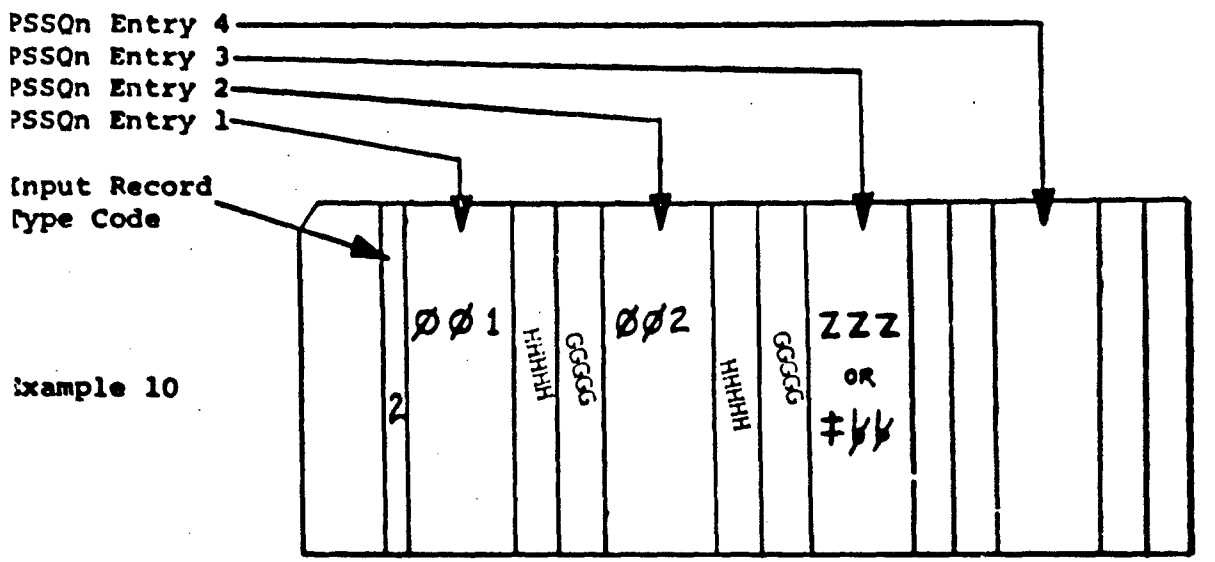
(a) This is the only time when the PSSQn in the input records can deviate from its normal (fixed) position.

(b) As previously mentioned, the PSSQn fields on the input record may be left blank if file maintenance is to assign the numbers. The PSSQn fields may contain pseudo numbers in it if the user wants more control over the order in which they are entered into the periodic set.

(c) It is not mandatory for a group of input fields comprising one subset to be immediately adjacent to a group of input fields comprising another subset.

(d) When there is room for a larger number of periodic subsets on the input record than the number that are actually entered, either place ZZZ in the next PSSQn field, or terminate the last periodic subset entry with a record mark, ‡, (0-2-8 punch).

For example: Assume that input record type 2 allows for 4 periodic set 1 periodic subset entries, but the data for only two subsets is to be entered.



(e) If the ≠ after the last subset is used, instead of the ZZZ entry, the subsets need not be in the order in which they appeared in the IDD.

NOTE:
A single subset can also be defined on an input record type. All the above rules pertaining to the multiple subsets will apply with the exception of the ZZZ and/or ‡ entry.

i. Variable Set.

(1) The variable set is commonly used as an unformatted remarks section used for commenting on the situation described in the other areas of the file record. It is usually textual in content and always variable in length. As such, there is nothing either fixed or periodic about its content and it is treated differently than these two file elements.

(2) Input to the variable set is the simplest of all inputs to design. From the standpoint of economy, it is best to reserve only one input record type exclusively for variable comments, and to use all available space on the input record for entering the variable information. The input record need only contain the following information:

(a) Input group control field.
(b) Input record type code field.
(c) Variable set data field (Input data for the variable set).
(d) External sequence number field (optional).

Items (a) and (b) have already been discussed, item (c) should be the maximum size available on the input record, after allowing for (a), (b), and (d).

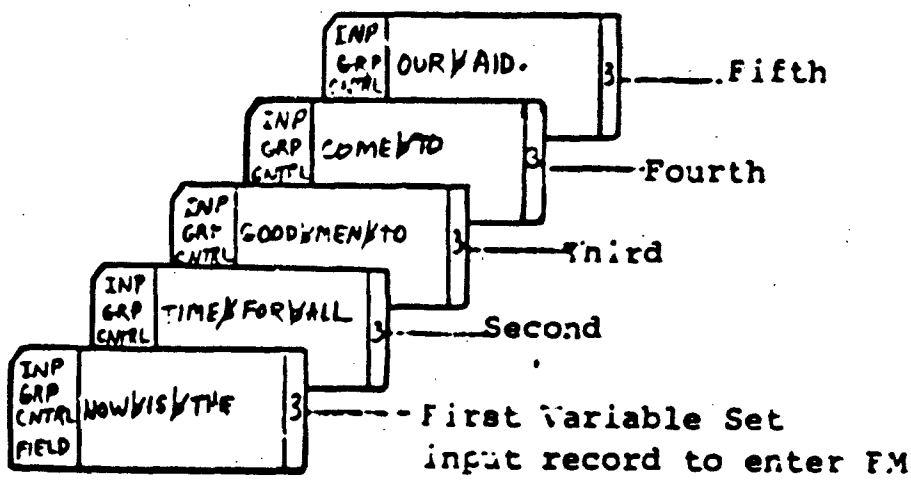
(3) Item (d) is an optional field that would probably never exceed 2 or 3 characters in length. It is used only for the purpose of sequencing the variable set input records (when on cards) within an input group. Since all variable set information is carried on the same input record type, a small catastrophe could result if these cards were dropped prior to input to file maintenance and there was no simple method to arrange them back into their original sequence within the input group. This external sequence number is not required by file maintenance and is therefore optional. Even if present, file maintenance does not examine it; it is up to the user to insure that the cards are in their proper order prior to their input to FM.

(4) The order in which the variable set input records enter file maintenance determines the order in which the comments are strung out in the variable set.

For example: Assume a variable set input record has been assigned input record type code 3.

ample 11

Input Group



The variable set in the file record would appear as:

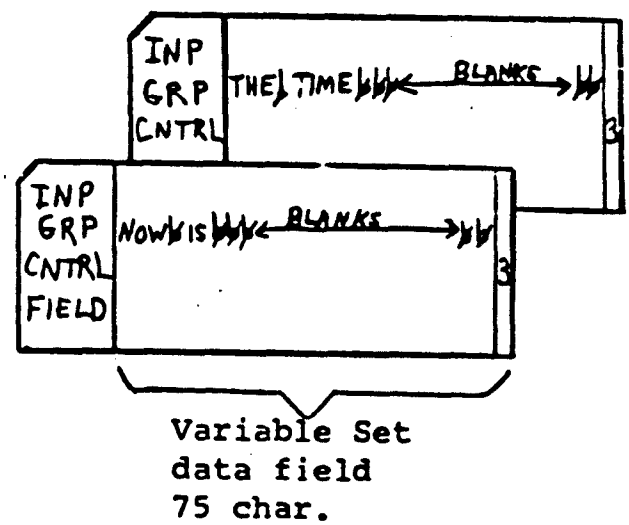
NOW∀IS∀THE∀TIME∀FOR∀ALL∀GOOD∀MEN∀TO∀COME∀TO∀OUR∀AID.∀

(5) When filling out (or key punching) a variable set field, the input analyst (or key puncher) should abide by the following rule: Always start and end each variable set data field with a complete word.

(6) Words cannot be hyphenated between input records. Any superfluous blanks left at the end of the variable comments field will be ignored by FM. FM always generates one space and places it to the right of the last word in the variable set data field.

For example: Assume a variable set data field 75 characters in length.

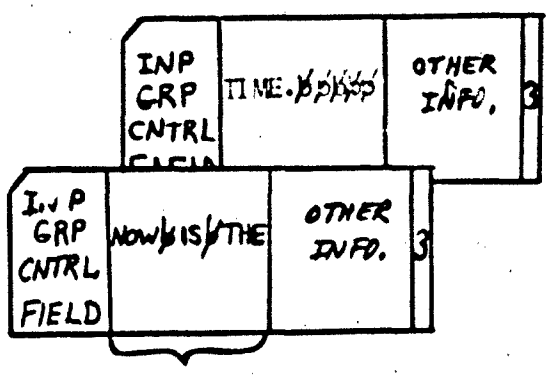
Example 12



The variable set in the file record would appear as: NOW∀IS∀THE∀TIME.∀

Example: Assume a variable set data field of 10 characters in length.

Example 13



10 char Variable
Set data field

The variable set in the file record would appear as: NOW IS THE TIME.

4-11. USING EXTERNAL FORMAT INPUT TO UPDATE FFS DATA FILES: External format input is useful for the routine updating and maintenance of existing FFS data files, as well as for producing an FFS data file from an existing (or created) file of data used as an E.F. input file. This subsection discusses information concerning the use of the external format input file for updating purposes as well as original FFS data file creation. It will lack some continuity unless the reader has some familiarity with the input descriptor deck specifications. There are two ways to approach the updating of data file using an external format input file. The input control card (ICC) opcode method is perhaps best suited to the user whose existing files of data do not fit the specifications for the (input) record opcode method.

a. ICC Opcode Method. An input file used for updating will, in all probability, contain data additions, deletions, changes, and creations. If the ICC opcode method is used, the input file must be separated by these functional responsibilities into 4 separate input files prior to processing. Each of these input files will then be associated with its own ICC and input descriptor deck, with the ICC indicating the appropriate operation for all input records of the input file (CAD,CMA,NEX, Add, Change, Delete, or Create). When the operation code is carried in the input control card (instead of the individual input records), it is known as an ICC opcode.

b. (Input) Record Opcode Method. If the designer of the input file is aware from the start that external format input files will be used for updating purposes, he should reserve one character on every input record type for the insertion of an operation code for the individual input record. This is called the record opcode. When the appropriate opcode is present in every input record, the entire input file can enter file maintenance as one input file, rather than having to be separated into separate input files by the operation to be performed.

c. Record Opcode Specifications. The location of the record opcode eld may vary from one input record type to another. It must occupy a ld by itself and its size is limited to one character. (It could also a zone bit inserted over an existing position in a numeric data field.) e record opcode symbols may be any alphanumeric characters and are fined by the user for each input file*. For example, the user could signate tha. for a particular input file:

+ = Add	N = NEX
C = Change	A = CAD
- = Delete	M = CMA
$ = Create	

e record opcode symbols must remain constant (in meaning) within an put file but different input files need not use the same record opcode mbols.

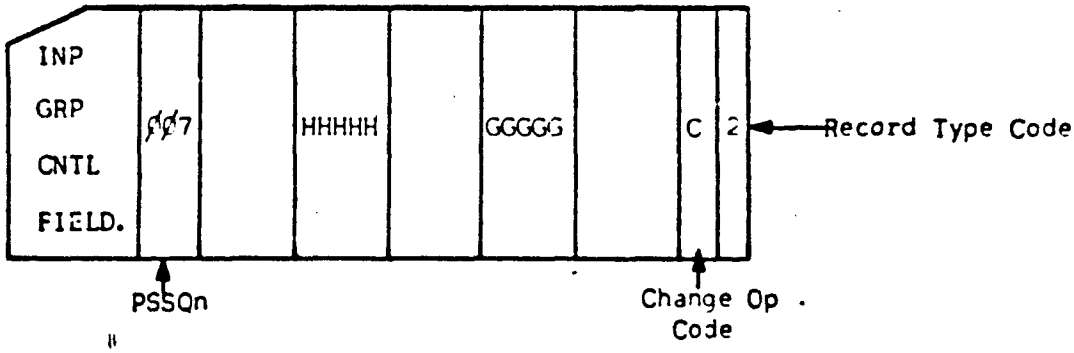
d. Limitations of Both Opcode Methods.

(1) Record Opcode Method Limitations.

(a) It should be noted from the above explanation of cord opcode symbols that each input record can carry only one record code. Therefore, that opcode will apply to every field, group, or bset that is to be extracted from that input record. (The data to be tracted from each input record type is specified by the field extract rds of the input descriptor deck.)

The use of "blank" for a record opcode is legal but not recommended.

ample: Assume input record type 2, designed to carry fields HHHHH and GGGG is used to change field GGGGG in subset 007.



(b) If the input descriptor deck specifies that both ields HHHHH and GGGGG will be extracted from a 2 type record, then:

1. The current file data for field HHHHHH must be entered on the 2 type record. If HHHHHH is left blank, then the blank field will be extracted and inserted in the data file as a legitimate change as well as the new GGGGG data, unless the "no process" blanks option has been specified on the input record type code locator card of the IDD. (The input record type code locator card is explained later.) or,

2. the IDD could be changed to specify the extraction of GGGGG fields only. This, of course, implies that all 2 type records in the input file will supply GGGGG fields only.

The person responsible for maintaining the data file may find that the updating of singular items in a file (i.e., a field, a group, a subset) is handled better by the internal format method.

(2) ICC Opcode Method Limitation. The limitations of the record opcode method also apply to the ICC opcode method. In addition to having only one opcode apply to all elements of an input record, that same opcode applies to every input record of the input file.

e. Multiple Usage of Input Record Elements. As defined by the IDD, any two or more of the following input record elements may actually be exactly the same, or partly consisting of the same elements of the input record.

Record ID field(s)

Input Control Group

Record Type Code

Record Opcode

Ordinary Data field(s)

4-12. SPECIFICATIONS FOR EXTERNAL FORMAT INPUT: External format (E.F.) input may be used to create or update more than one file data element for each input record. E.F. input data need not follow a rigid format as does I.F. input data. However, the E.F. input file must be preceded by an input descriptor deck (IDD) which provides a detailed description of the format of the E.F. input data records. Each IDD must be preceded by an input control card (ICC). The E.F. input file may be cards, card images on tape, or noncard image tape input in the form of IOCS Form 1, 2, 3, or 4 tape records (The forms of IOCS tape records are explained under the "E.F. input control card," figure 4-9.)

a. Specifications for the Input Control Card (for External Format). An ICC must precede each IDD to specify:

The file to be updated.

The input source and type.

The number of cards in the IDD.

The type and format of the input data, if tape input.

The opcode, when all input records are to have the same operation performed on them (ICC opcode method).

Whether or not the ICC and the IDD are to be listed.

The detailed specification of the ICC for external format is given in figure 4-9.

E.F. INPUT CONTROL CARD

USED ONLY IF TAPE INPUT IS SPECIFIED

1	2 6	7	8	9 10	11 13	14	15	16	17	18	19 22	23 26	71 72	73
Δ Card ID	FILE NAME	INPUT SOURCE	INPUT TYPE	IDD CARD COUNT	ICC OP CODE	PARITY	MODE	HEADER COUNT	RECORD FORM	PADDING CONSTANT	MAXIMUM BLOCK SIZE	MAXIMUM RECORD SIZE	SEQUENCE NUMBER	LIST FLAG
Δ	CMFLA	3	T	08	CRE	B	M	2	2	9	27000	09000	bb	L

column 01 = Card identifier Δ (11-7-8 punch). Indicates this is an input control card.
cols. 02-06 = Name of file being updated.
column 07 = Input source. Must be 3-9 for external format.
column 08 = Input type. "C" means input data is on CARDS.
"T" means input data is on TAPE.
cols. 09-10 = IDD card count. Number of cards in the input descriptor deck (should equal the sequence number punched in cols. 71 & 72 of the END card in the IDD).
cols. 11-13 = Operation code. If the same opcode is to be used for all of the input data associated with the following IDD; insert the opcode (ADD, CHG, CRE, of DEL) desired. If the opcode is contained in the input records themselves, leave this field blank. Figure 4-8 gives detailed information concerning the use of the opcodes. (If both ICC and record opcodes are used, the ICC opcode takes precedence.)

Columns 14 through 26 Are Blank Unless Input Data Is On Tape

column 14 = Parity of Tape. "U" for even parity.
"B" for odd parity.

Figure 4-9. The Input Control Card for External Format

Figure 4-9 (Continued). The Input Control Card for External Format

column 15 = Mode of Tape. "L" for Load Mode (wordmarks).
"M" for Move Mode (no wordmarks).

column 16 = Total Number of Header Records (including Tape Marks associated with the Header Records). Thus, if a header record and a tape mark precede the data, the number 2 would be entered in column 16.

column 17 = 1410 IOCS Record Form (1, 2, 3, or 4).
Form 1 - Unblocked, Fixed or Variable Length.
*Form 2 - Blocked, Fixed Length.
Form 3 - Unblocked, Fixed or Variable Length, with a character count field comprising the first four characters of the record. This field which specifies the number of characters in the physical record (block) is called the "Block Character Count."
*Form 4 - Blocked, Variable Length, with each logical record having a 4 character Record Character Count field, and the physical record beginning with a 4 character Block Character Count field. The Record Character Count field specifies the number of characters in the logical record.

*NOTE: Form 2 or 4 records must have each logical record terminated with a record mark.

column 18 = Padding Constant to be used (when required) to fill out the block, if form 2 records are specified. Nines (9's) are generally used for padding. The four special characters ✓ ≠≠* cannot be used for padding.

cols. 19-22 = Maximum Block Size, Maximum number of characters that may be encountered in a block, if form 2 or form 4 records specified. Maximum allowed is 9400, which may be used if the actual maximum block size is unknown.

cols. 23-26 = Maximum Record size to be encountered, unless Form 4 records are specified; in which case this entry specifies the relative L.O.P. of the Record Character Count Field. (For this purpose, the first position of a Form 4 record is considered 1, not 0.)

cols. 27-70 = Blanks.

cols. 71-72 = May either be blank or ØØ.

column 73 = Any non-blank character causes the ICC and the following IDD to be listed.

b. Specification for the Input Descriptor Deck. The input descriptor deck (IDD) provides a detailed description of the external format input records as well as designating the mode of handling each element of input data. The IDD (and the preceding ICC) must be on cards, even though the ICC specifies that the external format input file is on tape. The IDD may be thought of as consisting of two sections:

The Control Location Section.

The Field Extraction Section.

(1) Control Location Section. The control location section appears before the field extraction section. It identifies the location of various control information in the E.F. input records comprising the input file. The control location section may consist of up to four different types of cards:

Input Record Type Code Locator (Control Code Locator)

Input Group Control Field Locator ("S" Card)

Opcode Position Locator ("O" Card)

Record Control Field Locator ("C" Card)

The input record type code locator card is the first card in the IDD. This card is always required and there is only one per IDD. The second card of the IDD is the input group control field locator card. This card is also always required and appears only once per IDD. The next card in the IDD may be the Opcode position locator card, which is optional. There may be zero, one, or more than one Opcode position locator card(s) in the IDD. Next comes the record control field locator card, which is always required, and there may be more than one such card. The detailed specifications for each of these cards are given in the following pages.

(2) Field Extraction Section.

(a) The field extraction section of the IDD appears after the control location section. Its purpose is to identify the location of the data fields on the E.F. input records which contain the data to be used for creating or updating the data elements of the FFS data file. The field extraction section consists of field extract cards.

(b) An IDD END card is inserted after the last field extract card to signal the end of the IDD. Following the END card is the external format input file on cards; or, if the input file is on tape, FM begins reading tape unit A3 after reading the END card.

c. Input Record Type Code Locator Card Specifications. The input record type code locator is the first card of the IDD, immediately following

he ICC. This card, which is always required, gives the location of the ecord type codes in the input records. This card, which is always equired, gives the location of the record type codes in the input records. his card also allows the following two options:

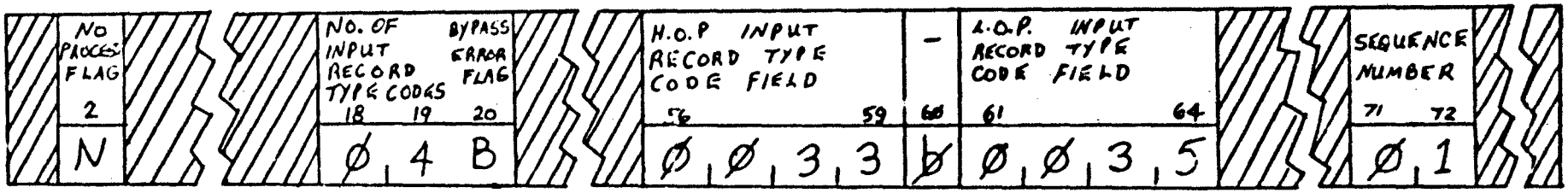
(1) By placing a N in column 2, those data elements of the input ecord which are blank will not be processed. (This may be utilized in elective processing of special fields in the input record.) If an N does ot appear in column 2, all of the data elements in the input record are rocessed as usual*.

(2) Placing a B in column 20 prevents the printing of an error essage and the input record when an input record contains a record type ode not specified in any of the field extract cards of the IDD. This rror bypassing may be required in the case where input data exists in ore than 30 different record type codes (the maximum definable in one DD) and the data is input twice, using a different IDD for each run. he absence of a B in column 20 allows "normal" printing of the error essage and input record when an input record does not have a record ype code specified in this IDD.

Figure 4-10 gives the detailed specifications of the input record type locator card.

*When using multiple subset entries the "no process" blanks option does not apply to the periodic subsets so entered.

INPUT RECORD TYPE CODE LOCATOR CARD



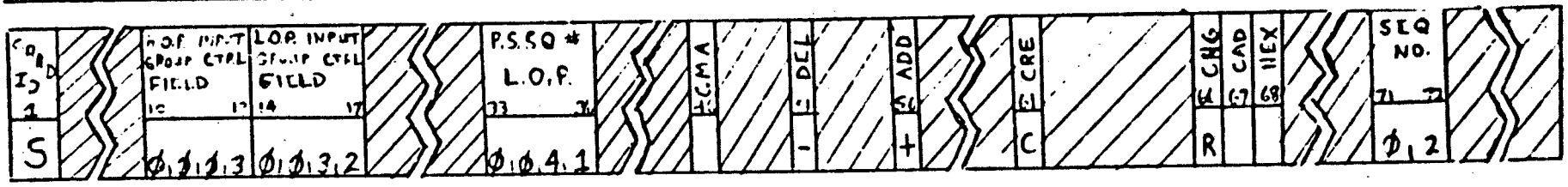
column 01 = Blank.
column 02 = N if blank data elements in the input records are NOT to be processed. Blank otherwise.
cols. 03 - 17 = Blank.
cols. 18 - 19 = Total number of input record type codes appearing in the field extract cards of this IDD.
column 20 = B if the printing of an error message, and the input record is to be bypassed, when the input record type code is not one specified by the field extract cards of this IDD.
cols. 21 - 55 = Blank.
cols. 56 - 59 = *Location of the high order (left most) position of the input record type code field, on the input records. For example, if a 3-character record type code were in columns 33, 34, and 35, of the input records (card input) this entry would be 0033.
column 60 = *Normally left blank, but for compatibility purposes may contain a dash (11 punch).
cols. 61 - 64 = *Location of the low order (right most) position of the input record type code field on the input records. In the example stated in columns 56-59, this entry would be 0035.
cols. 65 - 70 = Blank.
cols. 71 - 72 = Sequence number which must be 01.
cols. 73 - 80 = Blank.

*If the E.F. input file contains only one input record type and all input records are to be treated the same, leave cols. 56-64 blank.

Figure 4-10. The Input Record Type Code Locator Card.

d. Input Group Control Field Locator Card Specifications. The input group control field locator card is the second card of the IDD. This card gives the location of the input group control field on the input records. The purpose of the input group control field and the fact that it may be the same field as the record ID, has already been discussed. This card also gives the record opcode definitions and the PSSQ numbers if applicable. The detailed specifications for the input group control field locator card are given in figure 4-11.

INPUT GROUP CONTROL FIELD LOCATOR CARD



column 01 = S (Card Type Identifier)
cols. 02-09 = Blank
cols. 10-13 = Location of the high order (left most) position of the input group control field, on the input record. For example, assume that the record ID is in columns 3 through 32 of the input records and that this record ID is used as the input group control field. In this case the entry in columns 10-13 would be 0003.
cols. 14-17 = Location of the low order (right most) position of the input group control field, on the input record. In the example given in columns 10-13, this entry would be 0032.
cols. 18-28 = Blank
cols. 29-32 = Normally blank. But for compatibility purposes may be any characters. (These columns are ignored regardless of content)
cols. 33-36 = Location of the low order (right most) position of the three character "Periodic Subset Sequence Number" in the input record. If the input file does not contain any periodic data, or all of the periodic data it does contain is to have PSSQ numbers assigned by FM, leave these columns blank. NOTE: If the input record contains data for a periodic element and columns 33-36 are left blank, file maintenance will automatically assign the PSSQ number, if the operation is CREATE or ADD. If CHANGE or DELETE in this situation, an error message results and the transaction is not performed against the FFS data file.
cols. 37-45 = Blank
column 46 = *Record Opcode for CMA
cols. 47-50 = Blank
column 51 = *Record Opcode for DELETE
cols. 52-55 = Blank
column 56 = *Record Opcode for ADD
cols. 57-60 = Blank
column 61 = *Record Opcode for CREATE
cols. 62-65 = Blank
column 66 = *Record Opcode for CHG
column 67 = *Record Opcode for CAD
column 68 = *Record Opcode for NEX
cols. 69-70 = Blank
cols. 71-72 = Sequence number, which must be 02.
cols. 73-80 = Blank

Figure 4-11. The Input Group Control Field Locator Card

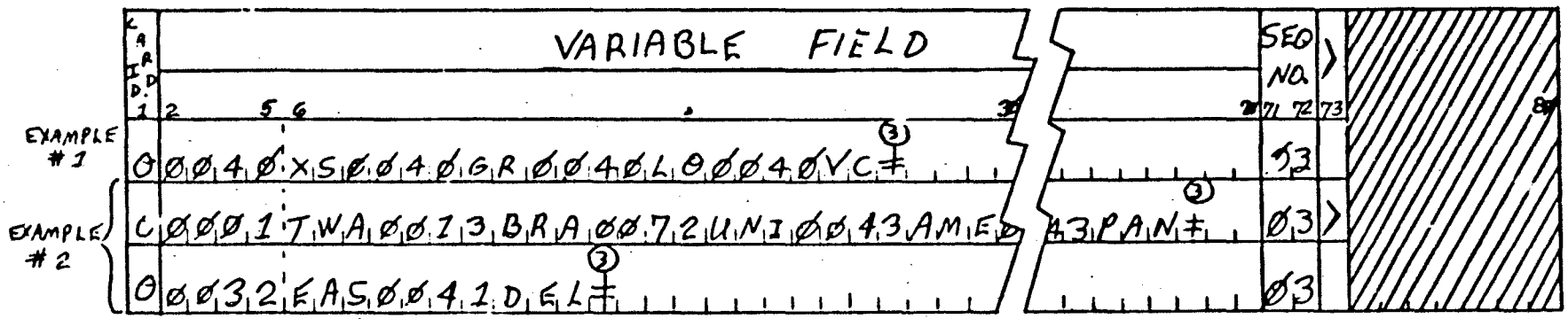
e. Opcode Position Locator Card Specifications.

(1) The opcode position locator card gives the position of the record operation code in the input records for each input record type. If there are 30 different record types specified by the IDD, then the opcode position locator card must specify 30 locations of opcodes, one for each record type code.

(2) If an ICC operation code (specified by the ICC) is used then there are no record opcodes used, and thus no need for specifying their position. In this case there is no opcode position locator card in the IDD.

Figure 4-12 gives the detailed specifications for the opcode position locator card.

OP CODE POSITION LOCATOR CARD



column 01 = Letter 0, Card type identifier.

cols. 02-05 = ✱Four digit Op Code Location.

cols. 06-XX = ✱Input Record Type Code. (See Note 1 and SPECIAL NOTE)

cols. XX+1-XX+4=

cols. XX+5-YY =

cols. etc. = (Note 2)

cols. =

cols. 71-72 = Sequence Number. Must be 03 or greater.

column 73 = > Continuation Symbol (6-8 punch) if entries require more than one card. Blank otherwise.

cols. 74-80 = Blanks.

SPECIAL NOTE: If cols. 56-64 of the Input Record Type Code Locator Card are left blank, then use a single blank as the input record type code on this card.

✱Repeat for each Record Type Code

Figure 4-12. The Opcode Position Locator Card

but will be a constant length within the same input file.

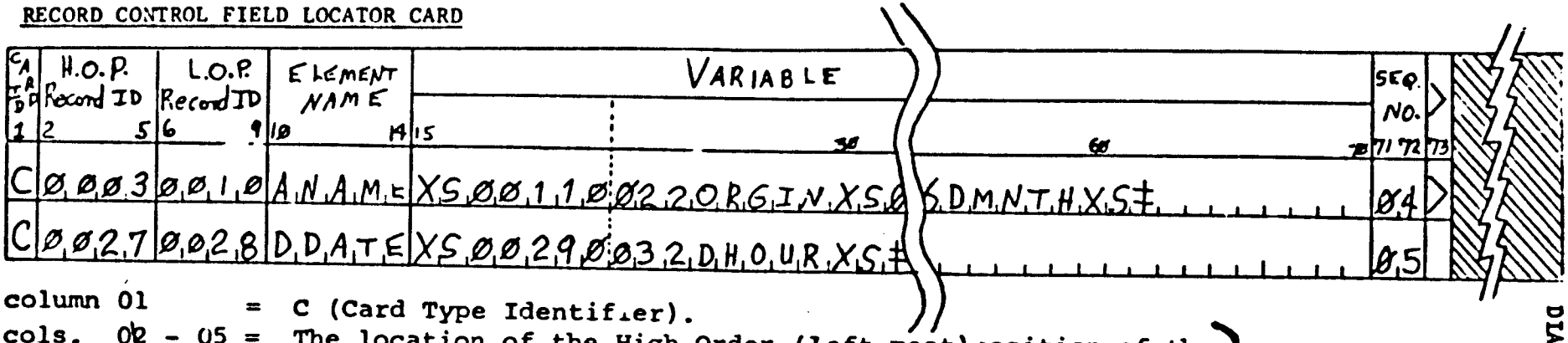
NOTE 2: The last entry on each opcode position locator card must be a complete entry (opcode location plus input record type code). This also insures that the first entry on a continuation card is the four digit opcode location.

NOTE 3: The last entry on each card may be terminated with a record mark ≠ (0-2-8 punch) although they need not be.

Figure 4-12, (Continued). The Opcode Position Locator Card

f Record Control Field Locator Card Specifications. The record control field locator card gives the location of the record ID in the input record and the record ID as it appears in the FFT. As explained earlier in this section (4-20), the record ID may be spread over several input record types within an input group. This card is required and there may be one or more continuations. The detailed specifications for the record control field locator card are given in figure 4-13.

RECORD CONTROL FIELD LOCATOR CARD



column 01 = C (Card Type Identifier).

cols. 02 - 05 = The location of the High Order (left most) position of the Record ID field in the input record.

cols. 06 - 09 = The location of the Low Order (right most) position of the Record ID field in the input record.

cols. 10 - 14 = Five character field name, as it appears in the FFT.

cols. 15 - XX = Record Type Code of input record in which the above field appears.

} Repeated for each field comprising the Record ID.

cols. XX+1-YY = Repeat entries, as stated for cols. 02-XX for each field comprising the Record ID.

cols. YY+1-ZZ = Repeat entries, as stated for cols. 02-XX for each field comprising the Record ID.

cols. 71 - 72 = Sequence number, next available. Cannot be lower than 03.

column 73 = Continuation symbol > (6-8 punch) if another Record Control Field Locator Card follows this one.

cols. 74 - 80 = Blank.

NOTE: Be sure to see additional comments concerning this card on the next page.

SPECIAL NOTE: If columns 56-64 of the Input Record Type Code Locator card are left blank, then use a single blank as the Record Type Code entry for each complete Record ID field entry on this card.

Figure 4-13. The Record Control Field Locator Card.

DIAM 65-9-1

g. Additional Comments Concerning the Record Control Field Locator Card.

(1) Continuation Cards - When continuation cards are required, do not split entries between cards. A complete entry is represented by the information in columns 02-XX, and if such an entry cannot be completed on a card it should not be begun, but instead given completely on the next card.

(2) The First Card - When there are continuation cards, insure that the first record control field locator card has no fewer entries than any of the other record control field locator cards. If necessary, re-arrange them and check that all of the cards except the last have the continuation punch in column 73.

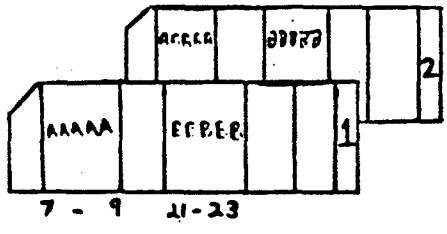
(3) Which Input Record Types To Specify - Many or all of the input record types may contain record ID fields. Only one of them need be specified by the record control field locator card, if the one chosen is a record type which will always appear in the input group. If there is no such record type, then enough (possibly all) of the input record types having record ID fields should be specified to insure that at least one of them will appear in any input group of the input file following this IDD.

(4) If the entire record ID is present in every input record, the user may either:

(a) specify the record ID fields to be extracted from a record type that will always be present in an input group; or

(b) specify the record ID fields to be extracted from every input record type if one or more types may be absent in the input group (by defining each field as many times as there are record types) - see the following example.

Assume AAAAA and BBBBB are record ID fields and present in every input record. The input group contains record types 1 and 2, either of which may be missing in the input group.



ie "C" card entries would appear as:

)0070009AAAAA1)0210023BBBBB100070009AAAAA200210023BBB

B2 ≠

f the record ID fields satisfy the following conditions, the record ID ields can be defined on the "C" card under the FFT group name.

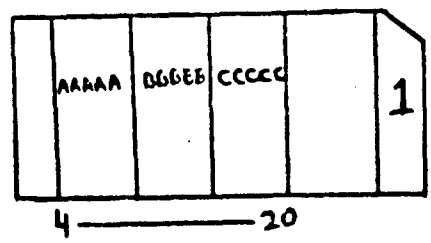
(a) The record ID fields have been defined in the FFT as a group.

ND (b) The record ID fields appear in the input record in the same sequence as they appear in the FILE (and adjacent to each other).

ND (c) None of the fields have to be processed by an input conversion subroutine.

cample: If the FFT contained the following entry for record ID fields AAAA, BBBBB, and CCCCC.

ECID GROUP AAAAA, BBBBB, CCCCC, and the input record appeared as:



ie "C" card entry could be defined in the following manner:

)0040020RECID1≠

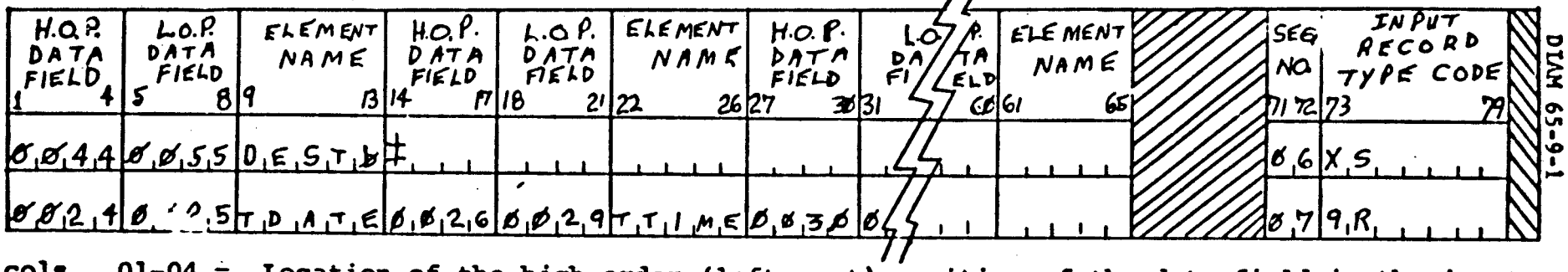
(5) Optional Record Mark - The last complete entry on each of ie record control field locator cards may be followed by a record mark ≠ (0-2-8 punch), although it need not be. This is allowed for compati- ility purposes.

h. Field Extract Card Specifications.

(1) The field extract card gives the location of those input ecord data elements which are to be extracted from the input record for se in maintaining the data file. Each field extract card refers to only ie input record type and contains that record type code, as well as the ame and location of the data element(s) to be extracted. All field xtract cards referencing the same input record type (i.e.: containing ie same record type code) must be together.

(2) The field extract cards appear in the IDD after all of the cards which comprise the control location section of the IDD. There should be at least as many field extract cards as there are input record types in the input file. The "Bypass" option of the input record type code locator card should only be used when there are more than 15 input record types in the input file, and two runs are made on the same E.F. input file with different IDD's. Unless this is the case, input records with undefined record type codes are recognized as being in error and are printed out with an appropriate error comment.

(3) The detailed specifications for the field extract card are given in figure 4-14.



cols. 01-04 = Location of the high order (left most) position of the data field in the input record.

cols. 05-08 = Location of the low order (right most) position of the data field in the input record.

cols. 09-13 = Name of the file record element for which the above data is intended, as it appears in the FFT.

cols. 14-26 = Same as for columns 01-13, if second element on this input record type. Blanks otherwise.

cols. 27-39 = Same as for columns 01-13, if third element on this input record type. Blanks otherwise.

cols. 40-52 = Same as for columns 01-13, if fourth element on this input record type. Blanks otherwise.

cols. 53-65 = Same as for columns 01-13, if fifth element on this input record type. Blanks otherwise.

cols. 66-70 = Blanks.

cols. 71-72 = Sequence number. Next available IDD sequence number.

cols. 73-79 = The Input Record Type Code for which this card specifies the fields to be extracted. (Left justified)

column 80 = Blank.

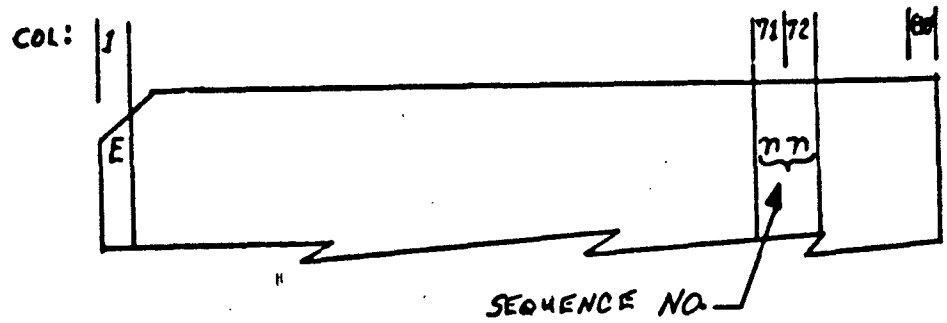
Figure 4-14. The Field Extract Card

NOTE: For compatibility, a record mark ≠ (0-2-8 punch) may immediately follow the last complete entry on a field extract card. (Column 14, 27, 40, 53, or 66 only.)

SPECIAL NOTE:	If the user does not want to extract data from a certain type input record which is present in the input file, the field extract card for that record type should still appear, but column 01 may be blank or a 0-2-8 punch. Columns 02-70 Blank and columns 71-79 should contain their normal entries.

Figure 4-14 (Continued). The Field Extract Card

i. (IDD) End Card Specifications. The End card signals the end of the IDD. The format of the End card is:



Column 01 = E (Card Type Identifier)
Columns 02-70 = May say anything if "E" is in column one.
Columns 71-72 = Sequence number. Next available sequence number. (This should be equal to the entry in columns 9-10 of the ICC for this IDD.)
Columns 73-80 = Blank

j. Additional Comments About The IDD.

(1) Multiple IDDs for input files. E.F. input files can be handled in a large variety of ways because an input file can be described by a number of input descriptor decks. This is of particular importance to the user who must build his file from existing files of data and who may find that the format of his existing files (to be used as the E.F. input file) conflicts with the requirements for input to FM. For example, assume that a certain input record type in the input file contains periodic information for two different periodic sets. According to the requirements previously specified, only periodic information for one periodic set can be present on any one input record type. The user can circumvent this problem by preparing two IDDs for the same input file. The first IDD, used during the creation of the file calls for only those periodic fields that belong to the same periodic set. The second input descriptor deck, used during the updating of the file, calls for the periodic information for the other periodic set.

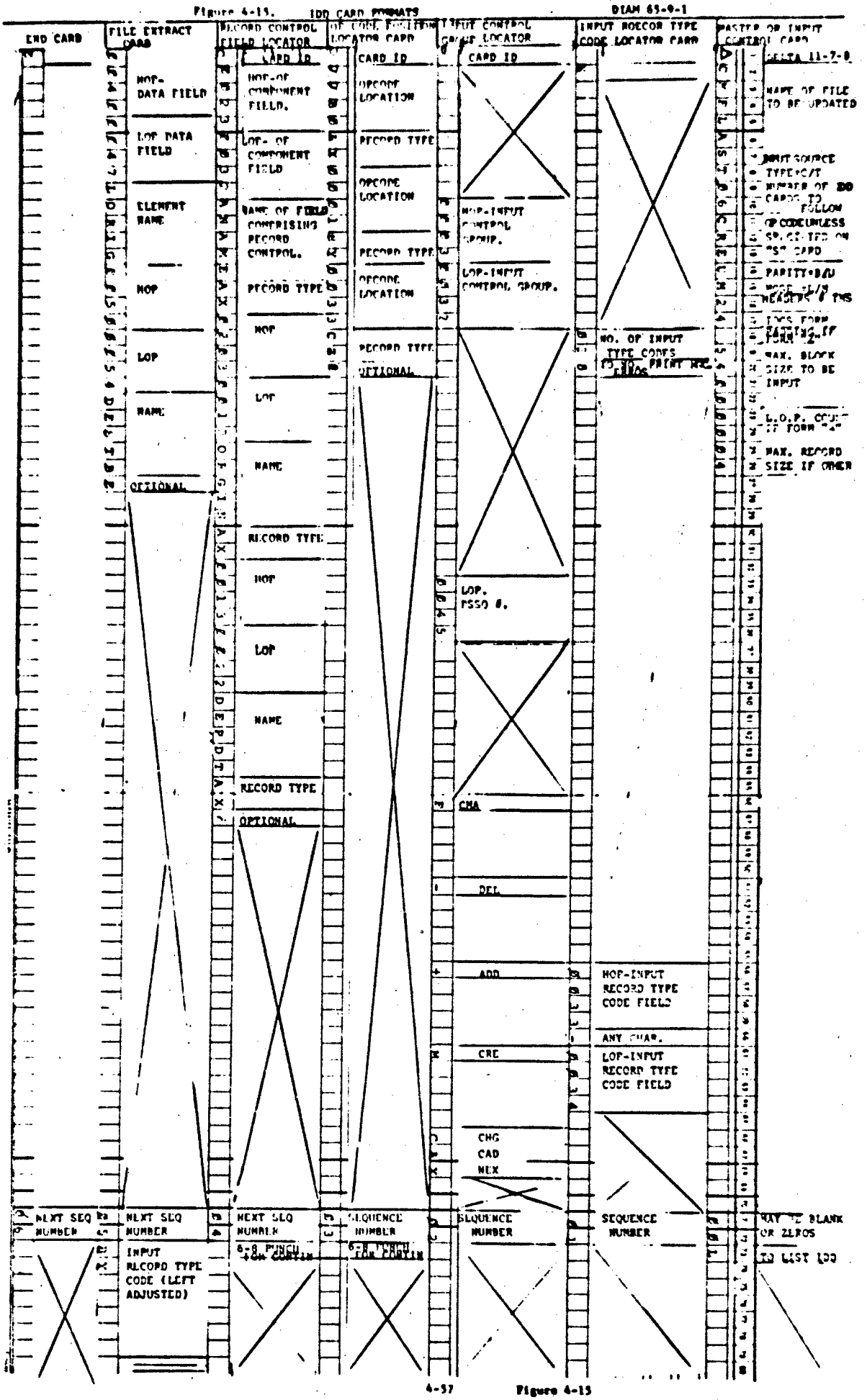

END CARD
FILE EXTRACT CARD
HOP-DATA FIELD
LOP DATA FIELD
ELEMENT NAME
HOP
LOP
NAME
OPTIONAL
RECORD CONTROL FIELD LOCATOR CARD
CARD ID
HOP-OF COMPONENT FIELD.
LOP- OF COMPONENT FIELD
NAME OF FIELD COMPRISING RECORD CONTROL.
RECORD TYPE
HOP
LOP
NAME
RECORD TYPE
HOP
LOP
NAME
RECORD TYPE
OPTIONAL
OF CODE FIELD LOCATOR CARD
CARD ID
OPCODE LOCATION
RECORD TYPE
OPCODE LOCATION
RECORD TYPE
OPCODE LOCATION
RECORD TYPE
OPTIONAL
TYPE CONTROL GROUP LOCATOR
CARD ID
HOP-INPUT CONTROL GROUP.
LOP-INPUT CONTROL GROUP.
LOP. PSSO 8.
CHA
DEL
ADD
CRE
CHG
CAD
NEX
INPUT RECCOR TYPE CODE LOCATOR CARD
NO. OF INPUT TYPE CODES TO BE PRINT MSG ERRORS
HOP-INPUT RECORD TYPE CODE FIELD
ANY CHAR.
LOP-INPUT RECORD TYPE CODE FIELD
SEQUENCE NUMBER
MASTER OR INPUT CONTROL CARD
DELTA 11-7-0
NAME OF FILE TO BE UPDATED
INPUT SOURCE TYPE+C/T
NUMBER OF IDD CARDS TO FOLLOW
OP CODE UNLESS SP. CF TED ON "S" CARD
PARITY+B/U
MODE 1/0
RECORDS IN THIS
INCS FORM FACTORS IF FORM "A"
MAX. BLOCK SIZE TO BE INPUT
L.O.P. CODE IF FORM "A"
MAX. RECORD SIZE IF OTHER
NEXT SEQ NUMBER
NEXT SEQ NUMBER
INPUT RECORD TYPE CODE (LEFT ADJUSTED)
NEXT SEQ NUMBER
6-8 PUNCH +1 on CONTIN
SEQUENCE NUMBER
6-8 PUNCH FOR PORTIN
SEQUENCE NUMBER
SEQUENCE NUMBER
MAY BE BLANK OR ZEROS
TO LIST IDD

Figure 4-15

65-9-1

(2) IDD Comment Cards. Cards with an asterisk (*) in column ay be used to identify the input descriptor deck for the user. These is may contain any comments in columns 2-80. They must not:

(a) Appear within the IDD.

(b) Appear between the ICC and the IDD.

(c) Appear between the IDD and the data.

s means that they may only appear immediately prior to an ICC. One ther restriction applies to the use of these comment cards. This is t if the ICC, IDD, and input file preceded by the comment card(s) form ob other than the initial job of the run, the input file of the previous must be ended with an IPPAKEND card (described on page 4-16). Normally IPPAKEND card is not required t) end the E.F. input file. The IDD ment cards are completely igonored by the system if they appear in the cified place.

3. THE TRANSACTION RECORD LAYOUT: The transaction record ha: been erred to many times in various sections of this manual. This is because is one of the basic elements in the file maintenance process. Figure 6 illustrates the layout of the transaction record.

a. General.

(1) The basic unit output by IP is the transaction record. In : case of internal format input, there is one transaction record gen- ıted for each input record. In the case of external format input there at least one transaction record built for each input record. If the :ration code is not DEL, the "TO DATA" field of the transaction record :ries the new data. If the opcode is DEL, the "TO DATA" field is blank. either case the "FROM DATA" field of the transaction record is blank :il it is filled by FM proper.

(2) The transaction records (and other data) on the trans- :ion tape output by IP are sorted by the fields indicated in figure 4-16 ajor to minor, left to right) by the FM sort phase. The transaction :ords from the sorted transaction tape are the input to FM proper and : used to maintain the file (create or update file records). While intaining the FFS data file, FM proper fills in the "FROM DATA" field of :h transaction record with the old data (unless the opcode is CRE). proper flags any transaction records in the RELHO (relative HOP of ta) field which are in error, and cannot be used. These flags and their anings are listed later in this section, under "Transaction Confirmation sting - Error Flags."

(3) The transaction records, now in their final state, are tput by FM proper onto the transaction confirm:ion tape. This tape is en printed by the output execution phase of output processing to provide record of file maintenance performed.

(4) The following table gives the source of each element com- ising the transaction record, and a brief description of the element.

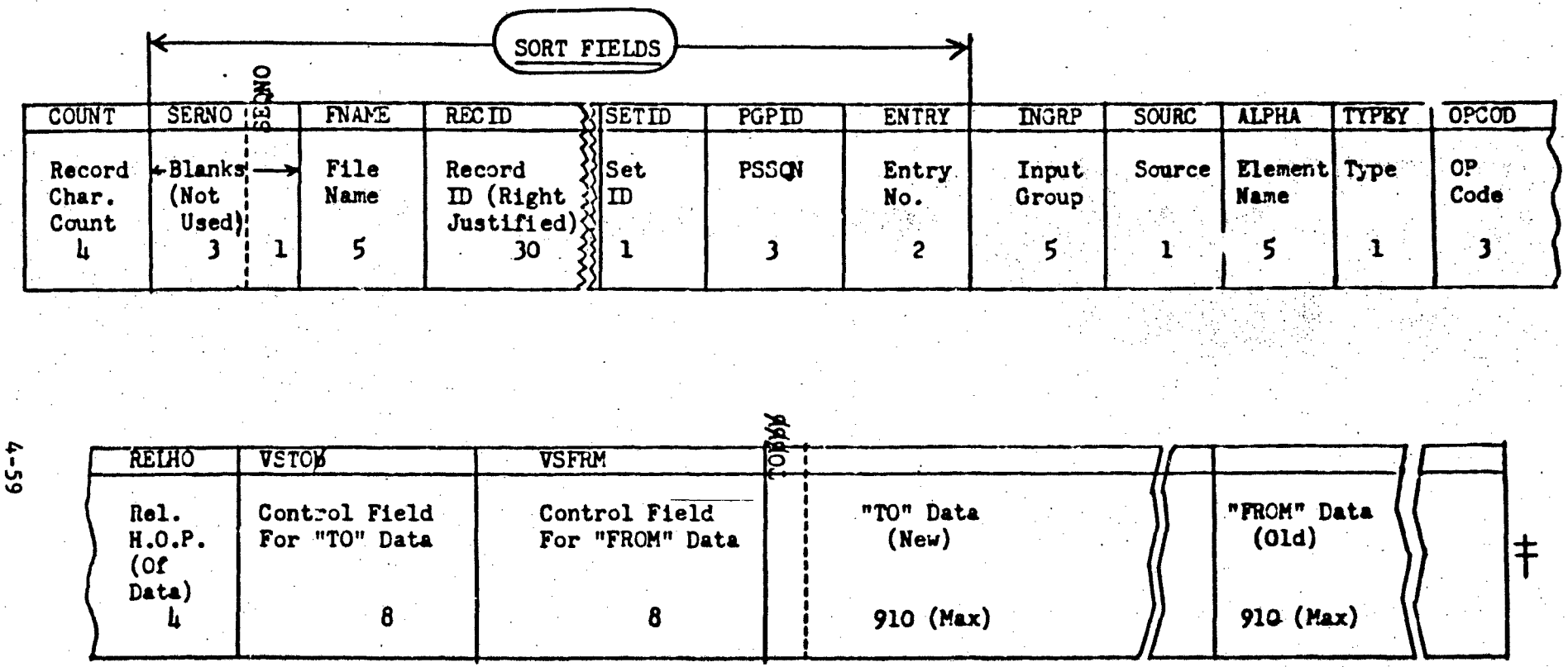


Figure 4-16. Transaction Record Layout

FILE MAINTENANCE TRANSACTION RECORD ENTRIES

:MONIC	SOURCE	DESCRIPTION
:NT	IP	Record character count of Transaction record.
RNO	N/A	Blanks.
QNO	N/A	Blank.
AME	ICC	Five-character file name.
:ID	Input File	Record ID field (Record Control Group) as carried in FFS data file record. This field is Right justified.
TID	FFT	Set ID as specified in LR #2 of the FFT: X = Fixed Set 1 - 8 = Periodic Sets 9 = Variable Set
PID	Input File or FM	Periodic Subset Sequence Number. If not provided in the input record, will be assigned artificially by FMIP.
TRY	FFT	FFT entry number.
GRP	IP	Five-character Alpha field giving the five data characters contained in the LOP of the Input Group Control Field. If this field is less than 25 the INGRP field is blank.
URC	ICC	Source Code: 1 = Retrieval 2 = Internal Format 3 - 9 = External Sources
PHA	ICD or Internal Format Input Record	Five-character mnemonic of the field/group/subset contained in the Transaction Record.
YPE	FFT	Type code as contained in LR #2 of the FFT: 1 = Control Field/Group 2 = Fixed Field/Group 3 = Periodic Field/Group 4 = Variable Set 5 = Periodic Subset 6 = Periodic Set Control 7 = Variable Set Control

MNEMONIC	SOURCE	DESCRIPTION
OPCOD	ICC or Input Record	Operation code for this transaction: CHG = Change ADD = Add DEL = Delete CRE = Create CMA = CHG, ADD CAD = CRE, CHG, ADD NEX = No. Op.
RELHO	FFT; or if error, FM Proper.	Relative high order position of the transaction item in the FFS data record; or if error, contains the error flag.
VSTOb	IP	Variable set control for the "TO" portion of the transaction record. Used by OP when listing transactions.
VSFRM	IP	Variable set control for the "FROM" portion of the transaction record. Used by OP when listing transactions.
TObbbb	Input File	Starting location of the "TO/FROM" area of the transaction record. Depending on the individual transaction, there may be a TO entry, TO and FROM entries, or neither. If TO entry is present, it will contain change data or a message. If FROM area is present, it will contain blanks and be the same size as the TO area.

(5) The transaction record is variable in length with a maximum size of 1004 positions. The records will be blocked in accordance with IOCS requirements for Form 4 records, each block preceded by a block count.

b. Transaction Confirmation Listing.

(1) Two RIT's have been placed on the FFS History File supplied to the user for the purpose of printing the transaction confirmation tape. They are:

FMRFR - The function of this RIT is to print all transaction records processed by file maintenance. Erroneous transactions will be flagged with an error code.

FMRDR - The function of this RIT is to print only the transaction records found to be in error.

(2) These RIT's were written in AUTOCODER because certain limits of the report structuring phase of the output program were exceeded. They are to be assembled and placed in the FFS Relocatable Execution Library as all other RIT's.

(3) To print the transaction confirmation tape, place the following cards immediately after the file maintenance deck, but before the MON$$ END card.

```
Column      1    6          16    21
                 MON$$      EXEQ  OP
            SOURCE FILE MAINTENANCE
            CLASS (cols. 7-36 contain classification)
            PUBLISH FMRFR (or FMRDR)
                 MON$$      END
```

(4) If the tape is being printed at some later date, mount it on tape unit B1 and use the above cards with the standard ASGN cards for the output program.

4-14. INPUT PROCESSOR PHASE ERROR MESSAGES:

a. IP Error Specifications

(1) IP keeps a count of the number of valid transaction records put out and a count of the number of error transactions (which are not put out). At the end of each job and/or at intervals of 500 input records, IP compares these two counts. (As used here a "job" begins with each IDD.) If the number of errors is greater than 30% of the number of valid transaction records put out, the job is deleted, and any transaction records already put out by this job are logically destroyed. The next job (if any) is then processed.

(2) Each such job deletion causes the printing of error messages on the console typewriter plus the following standard job messages on both the 1403 printer and the console typewriter:

PROCESSING SOURCE N, DATA FOR FILE XXXXX

NNNNNNNNN TRANSACTIONS NNNNNNNNN DATA ERRORS

(3) A summary of the various errors detected by the IP phase follows, listed under five error procedure headings.

(a) Delete Entire Run (System Errors)

The FFT2 entry for a data field or a control field lacks a require parameter.

A conversion subroutine takes the system error exit.

(b) Delete Job

ICC errors

ICC not found
FILE name incorrect
Source code error
Illegal opcode
Incorrect IDD card total
Illegal input tape parity designation

IDD errors

Missing card in IDD
Incorrect sequence number
Field name error
Input field's location in input record implies it's too large for FFS data file
Relative field location specified is larger than the input record
Too many input record types (>30) described

Subroutine too large (exceeds 9999 characters)

Too high a proportion of errors.

(c) Delete Field from Input Format Table

Required subroutine not in FFS Library directory

Field wrong type for ICC opcode

(d) Do Not Process Input Record But Print It*
*(Will not generate a transaction record)

External format input file errors

Record type code error
Record opcode error

Internal format input data deck

Illegal opcode
Wrong size record ID
Incorrect data field name
Input data field too large for element in FFS data file
Required subroutine for field not in FFS Library directory

(e) Do Not Process This Data But Print It Out (With Transaction Record)

Wrong type field for record opcode (e.g., change opcode against variable set)

Bad data (subroutine took data error exit)

Illegal (blank) periodic subset sequence number (e.g., PSSQn blank with a DEL opcode)

(f) Input Processor Error Comments Listing, With Explanations. The following four pages list all of the error comments which IP may print on the 1403 printer. The list is in alphabetical order. Those few messages whose alphabetical order is indeterminate are listed first. It should be noted that the comments are easier to interpret when accompanied by supporting data and input record listings than they may appear to be in this compilation.

X - CARD MISSING. (The specified card is missing from the IDD. "F" for field extract, "S" for input group control field locator, "O" for opcode position locator, "C" for record control field locator.)
XXXXX - HAS HIGH AND LOW ORDER POSITIONS REVERSED (HOP specified for XXXXX is bigger than LOP. Correct the appropriate card in the IDD.)
XXXXX IS RECORD CHARACTER COUNT FIELD. (The IDD has specified some operation upon the record character count field, which is illegal.)
CARD XX READ. CARD XX REQUIRED. (Card sequence error in the IDD. Check for misplaced, missing, and/or extraneous cards in the IDD. Also check for mispunched sequence numbers.)
CONTROL FIELD XXXXX HAS Y AS FIELD TYPE. (The control field specified in XXXXX, is not listed in the FFT as a control field, but as Y.)
CONTROL FIELD XXXXX LACKS PROPER FFT2 PARAMETER. (LR2 of the FFT for this file is in error concerning the control field/group specified in XXXXX. The FFT must be properly structured and placed on the FFS Relocatable Execution Library with the field specified as Numer or Alpha. Detection of this error causes IP to delete the entire run.)
CONTROL FIELD NAME XXXXX, NOT IN FFT. (If it is not obvious that the wrong control field name has been used, check it against the FFT for the file.)
DATA CHECK. (The ICC may have specified the wrong parity for the external format input data tape. However, this could also be due to a permanent read error on the input data tape for which correct parity was specified. 500 such errors cause job deletion.)

DATA FIELD XXXXX TOO LARGE, SHOULD BE nnnn.	(If error in field size is not obvious, check the size of the element as specified in the FFT, against the size input to FM.)
DATA FIELD NAME WRONG	(The name of the element specified in the I.F. data input card is incorrect.)
ERROR PERCENTAGE TOO GREAT	(The count of errors has exceeded 30% of the count of valid transaction records output. Job is abandoned.)
FIELD XXXXX LACKS FFT2 PARAMETER	(LR2 of the FFT for this file is in error concerning the element (group or field) specified in XXXXX. The FFT must be properly structured and placed on the FFS Relocatable Execution Library with the field specified as either Numer, Alpha or requiring subroutine conversion.)
FIELD NAME XXXXX NOT IN FFT	(The element name specified in XXXXX has been used in the IDD, but is not listed as a field, group, or subset name in LR3 of the FFT for the file. If it is not obvious that the wrong element name has been used, check it against LR3 of the FFT for the file.)
FILE NAME NOT IN DIRECTORY	(File name specified in ICC, has no FFT. Probably wrong name in ICC, but could mean that FFT for file has not been put onto the FFS Relocatable Execution Library.)
FIRST CARD OF IP PACK MUST BE THE INPUT CONTROL CARD - △ IN COLUMN 1	(11-7-8) (ICC missing or out of place. Note: the symbol △ will not be printed by the 1403.)
IDD CARD TOTAL INCORRECT	(The number of cards specified for the IDD in the ICC is wrong OR the IDD has too few or too many cards.)
IP TABLE EXCEEDED	(Too many different field extract cards and/or too many input record type codes (30) specified max. in the IDD.)

OPCODE X NOT GIVEN IN IDD	(The record opcode specified is in the input file, but was not given on the input group control field locator card of the IDD. Correct either the input file's opcode or the IDD.)
OP DOES NOT APPLY TO THIS TYPE FIELD (X)	(Either this field cannot be operated upon by the specified opcode, OR the wrong data element name has been specified in the IDD if E.F. input, or the I.F. data card if internal format.)
OP ILLEGAL	(Illegal operation specified in ICC, or internal format data card. Check spelling of operation.)
PERIODIC GROUP ID MISSING	(PSSQn LOP location incorrect or not given in input group control field locator card of IDD, when it should have been, OR an actual periodic subset sequence number is required, but the PSSQn field on the input record is blank or a pseudo number.)
RECORD ID WRONG SIZE	(The record ID specified in the I.F. input data card is either too short or too long. Maximum length is 30 characters. Minimum length is 1 character.)
RECORD TYPE CODE XXXXXXX NOT IN IDD	(The record type code specified is in the E.F. input file (in the position specified by the input record type locator card) but was not given in the field extract card(s) of the IDD. Correct either the E.F. input file or the IDD. It is also possible that the opcode locator card doesn't specify this record type code.)
RECORD TYPE CODE FIELD TOO LARGE	(Seven characters maximum.)

SOURCE NOT IN FFT2 FOR XXXXX	(The source specified in the ICC for the specified element, or the name of the element is wrong, or FFT has been mis-structured. LR2 of FFT should contain a source code for each element.
SUBROUTINE XXXXX, IS TOO LARGE (MAX = XXXXX)	(The specified input conversion subroutine must be shortened. It exceeds the size of the IP subroutine area available for this file.)
SUBROUTINE XXXXX, REQUIRED FOR FIELD YYYYY, NOT IN DIRECTORY	(The input conversion subroutine specified for XXXXX must be in the FFS Relocatable Execution Library in order to process the input data for field YYYYY, but it isn't.)
SUBROUTINE XXXXX TOOK DATA ERROR EXIT	(The specified input conversion subroutine finds the input data to be in error. Compare data with what subroutine expects it to be like.)
SUBROUTINE XXXXX TOOK SYSTEM ERROR EXIT	(The specified subroutine took the system error exit due to either the file name or data name not being in the subroutine system check table. Upon detection of this fact, IP deletes the run.)
THIS FIELD XXXXX, EXTENDS BEYOND RECORD BOUNDS	(Data element specified has a LOC specified which is greater than maximum input record size, as given in the ICC.)
TOO MANY FE-CARDS	(Too many field extract cards in the IDD or the number of record types given in the input record type code locator card of the IDD is incorrect.)

4-15. TRANSACTION CONFIRMATION LISTING - ERROR FLAGS:

a. The transaction confirmation listing is made when output processing is used to print out a transaction confirmation tape. When FM proper is generating the transaction confirmation tape, it enters a four-character error flag into the "relative high order position" (RELHO) field of any transaction record in error. The meaning of each of these flags is given below.

b. If any of the transactions were in error they will be noted on the listing by a message:

XXXXXXXXXXX **ERROR**

This message accompanies the listing of the transaction record itself. Transactions noted in this manner HAVE NOT BEEN PERFORMED. They must be corrected and reentered.

BLAN - Blank Record Control

The record control group (record ID) for the transaction is blank.

CREA - No Such Record ID In File

In comparing record ID's, FM proper failed to find an existing file record whose record ID matches that of the transaction record. Such a condition would be valid for only a CREATE transaction, and the transaction in which this flag appears is either a CHG, DEL, or ADD.

DBLA - Data For Same ID As Previous Transaction

The transaction is for the same field in the same subset in the same set of the same record as a preceding transaction, or else an attempt was made to add a group, after a field contained in that group had just been added.

ERRØ - Data Error Flagged During Input Processor Phase

Input data failed validity check in IP. This can be caused by subroutine error exit following attempted conversion.

MAXL - Add Would Exceed XXXX* Characters

The limit on the length of a data record is XXXX* characters. This transaction is an ADD which, if it were performed, would result in the record exceeding that limit.

*Note - May be unique to Site. (The "standard" maximum size is 5,400.)

NADD - Not Added - Subset Already Exists

The transaction specified an ADD of a field/group or periodic subset, and it was found that there already exists in the file record a periodic subset whose periodic subset sequence number is identical to that of the transaction. This error is probably caused by an attempted ADD of a field/group to a periodic subset which exists but apparently does not contain data in the specified field/group. Such an operation must be entered as a CHG since in reality the particular field/group does exist, but is composed entirely of blanks.

NCRF - New Record Being Created, Opcode Not CRE or CAD

User attempted to update to periodic field with ADD opcode. The opcode must be CHG, CAD, or CMA.

NOCR - Not Created - Record Already Exists

The transaction specified a CREATE, but an existing file record was found with the same record ID as the one which was to have been created. It is impossible to have a CRE transaction against an existing file record. For instance, if you want to add a new subset to an existing record, you must use an ADD since a CRE is valid only for an entirely new record.

NOID - No Such Subset In Record

In the current file record there is no subset with a subset sequence number which matches the one specified in the transaction. This condition would be valid only for the ADD of a subset, while the transaction is either a CHG or DEL.

NOOP - Invalid Operation Code

The operation field in the transaction contains an invalid entry. The only valid entries are CRE, ADD, DEL, and CHG.

NSET - No Such Set In File

The Set ID in the transaction is not one of those specified in the FFT.

NTYP - Invalid Type Entry Code

An illegal type of transaction was attempted. For example, CHG against the variable set.

NVST - Update to USET Not An ADD

User attempted update to USET; the opcode specified was not ADD.

PSER - Transaction Precluded By Above Set Delete

If, in a batch of transactions for the same record ID, there is specified a delete of a periodic set, there cannot be in that same batch any transactions against any periodic set with a lower set ID than the one being deleted.

RCTL - Record ID Error Flag From IP

User attempted illegal operation on record control.

SORT - Transaction Record Out of Sequence

This transaction on the transaction tape is out of sort or an invalid record control in some transaction has caused an "out of sort order" condition to appear to exist.

SSID - Incorrect Subset ID In Data

In the internal format CHG or ADD of an entire periodic subset, the first three characters of the data field comprise the periodic subset sequence number. Therefore, for the CHG of an entire existing subset or the ADD of an entire subset with an externally assigned (actual) subset sequence number, the 3 high order data characters must be identical to the "periodic subset seq. no." in columns 11-13 of the internal format input form. For an ADD of an alpha subset with a pseudo PSSQn, the 3 high order data characters must be blank. If it is a numeric subset, then 3 zeros must occupy the high order positions.

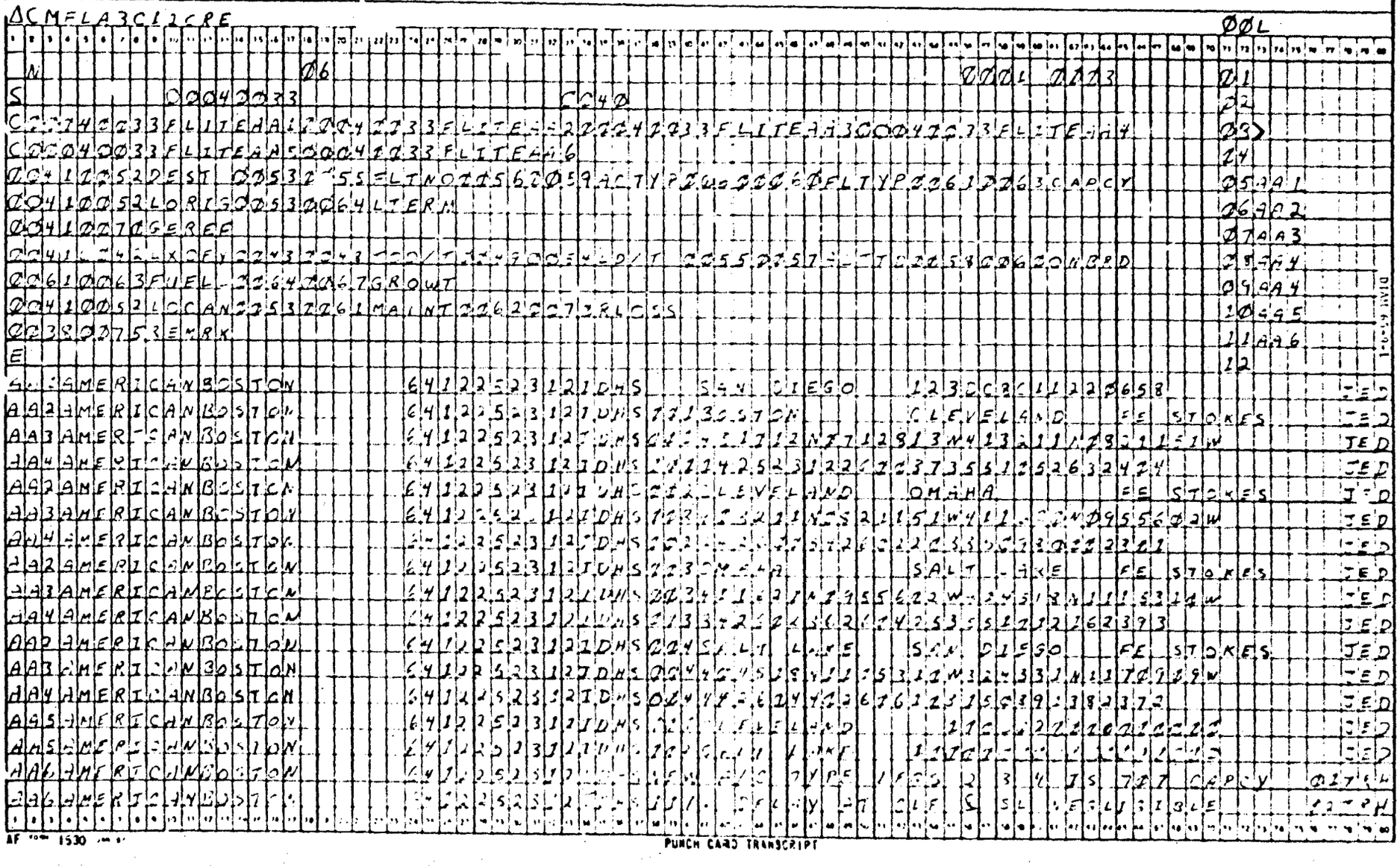
4-16. CROSS-INDEX UPDATER PHASE ERROR COMMENTS LISTING, WITH EXPLANATIONS: The following is a list of all the error comments which CIU may print on the 1403 printer.

NOTE: None of the following errors being detected causes the program to stop. Each error printout will be accompanied by the record ID of the record involved in the error.

UPDATER ERROR LIST	DATE "nnnnnn" ERROR FOUND (This is a CIU error heading that precedes each CIU error printout.)
INVALID ADD FUNCTION EXISTS IN CROSS INDEX	(There is an error in the format of cross index, i.e., the record ID to be added cannot be found.)
UNABLE TO PROCESS FOLLOWING RECORD	(There is an error in the format of cross index, i.e., record ID to be deleted cannot be found.)
WRONG TRANSACTION CHARACTER. TRANSACTION IGNORED	(This printout occurs when CIU is not in CREATE mode and the cross-index transaction character is neither ADD nor DELETE. Probably due to a blank or wrong transaction character.)

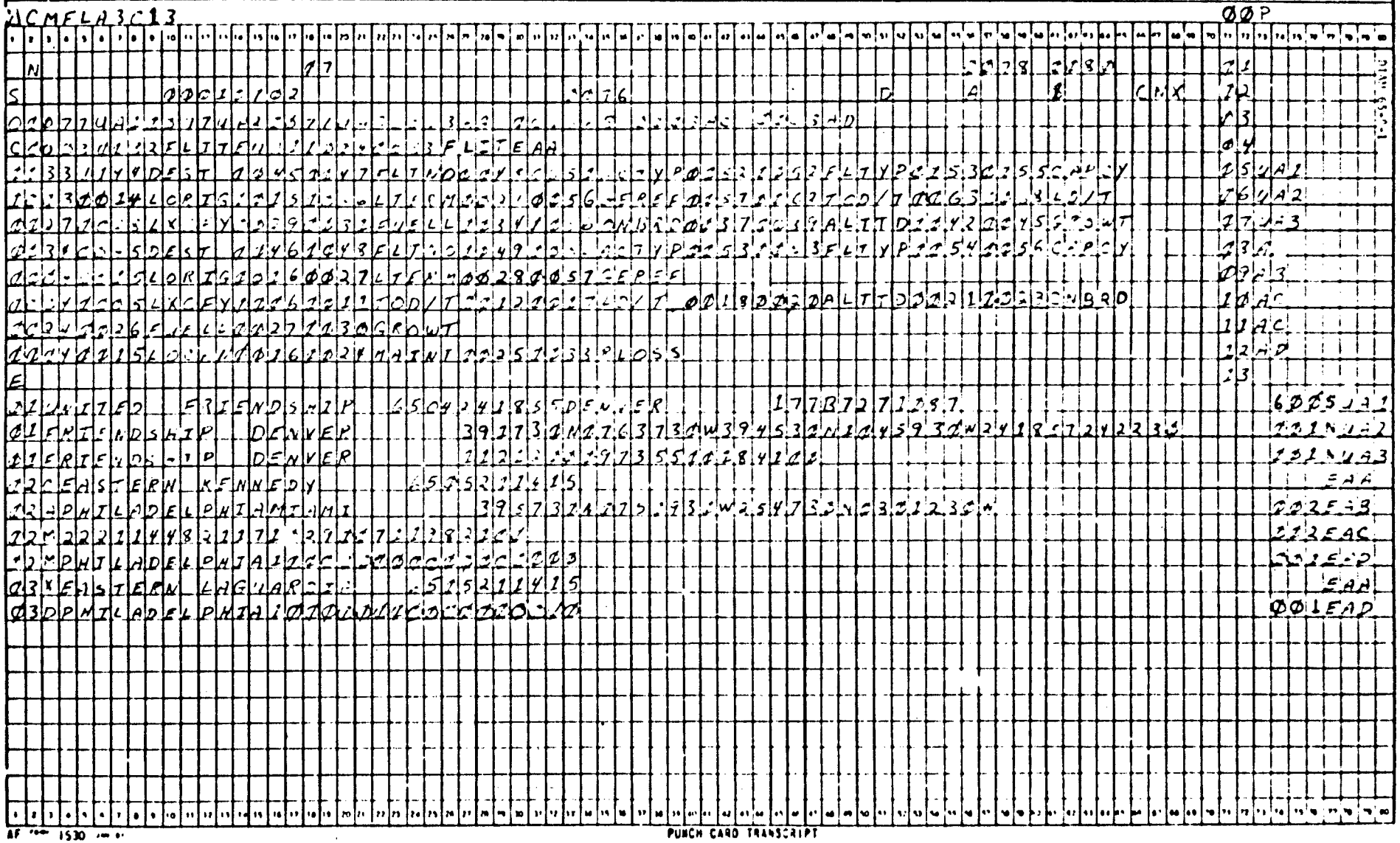
4-17. EXTERNAL FORMAT INPUT EXAMPLES USING THE CMFLA SAMPLE FILE: The following three examples (figures 4-17, 4-18, and 4-19) illustrate the use of external format input to maintain elements of the CMFLA sample file. They are purposely kept simple to more clearly show the interrelationship between the ICC, the cards of the IDD, and the input records.

Figure 4-17. External Format Input Sample Example #1



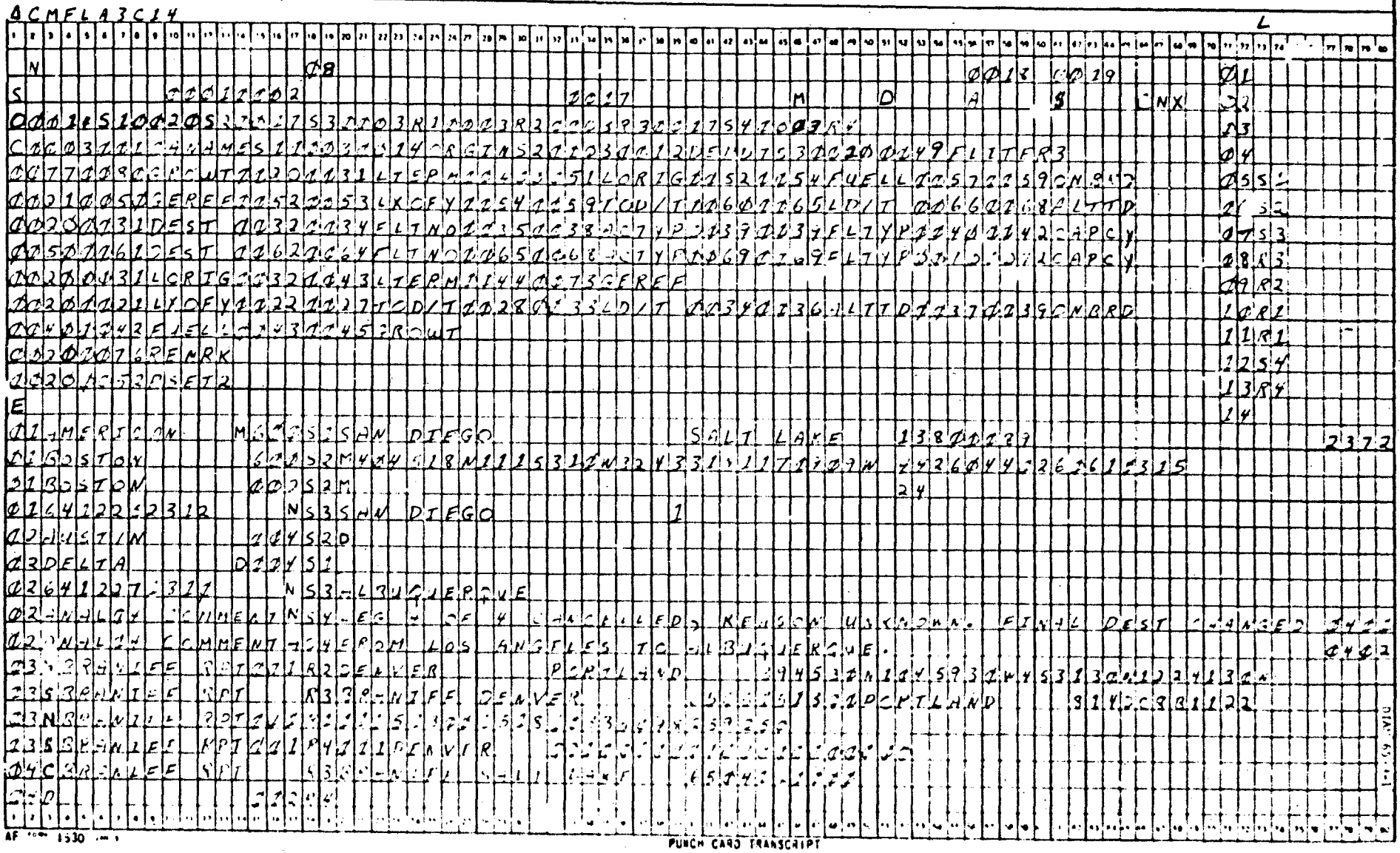
PUNCH CARD TRANSCRIPT

Figure 4-18. External Format Input Sample, Example #2



PUNCH CARD TRANSCRIPT

Figure 4-19. External Format Input Sample, Example #1



PUNCH CARD TRANSCRIPT

LOGICAL FILE MAINTENANCE

4-18. WHY LOGICAL MAINTENANCE? Why should an analyst use the capabilities of LFM? This seems a simple question, yet cannot be simply answered. In some cases, data present in a file can be manipulated with less effort by the use of logical maintenance, as opposed to doing the same job with a standard maintenance run. In other instances, LFM may provide a tool for reducing the amount of data necessary as input for the creation of a data record. Use of LFM capabilities might be required in order to reduce input errors, or to "automatically" update critical fields of subordinate files with data from a "master" file. Following are descriptions of required control cards, a series of examples of the use of Logical Maintenance, and the reasons for the selection of LFM to do the job.

4-19. SUBSYSTEM OPERATION:

a. Formatted File System records are logically updated in a "file in - file out" manner. Execution programs are generated through autocoder compilation and placed in the system for subsequent execution. Logical maintenance execution programs (those programs that actually update files) are maintained in the program segment of the FFS, identified by an L in the first character position (LMB1P). If lookup tables are required by an execution program, they are stored in the subroutine segment of the system, also identified by a leading L (LMBAS, LMMBS, etc.). A two-character mnemonic (MB) serves to differentiate one program from another and to link programs with their subroutines (tables). Thus, the LFM job "MB" might generate the program LMB1P and tables LMBAS, LMBBS, LMBCS, and LMBDS. When execution of this job is required, the LFM supervisor calls in the correct program and associated tables, builds a master table directory, and turns control over to the execution program. When a table lookup is required the execution program returns to the supervisor which determines what table is required, calls in the proper table, performs the lookup, indicates the result, and returns to the execution program.

b. At present, one control card is necessary to indicate to the LFM supervisor the nature of the job required. This control card contains "LMJOB" in card columns 1-5, a two-character job name in columns 7-8, and the type of job beginning in column 10. These entries in column 10 are:

COMPILE - Indicates that an LFM execution program must be selected, compiled, and added to the FFS.

CREATE - Indicates that master lookup tables and an execution program must be compiled and added to the FFS. Only used when job is in LOOKUP mode.

EXECUTE - Indicates that an LFM execution program previously added to the FFS must be run.

4-20. CONTROL CARD ENTRIES:

a. The LFM system was written under the assumption that the prospective user would have a passing aquaintance with the FFS retrieval and output language.

b. Control card types must begin in column 1. Each entry must be separated from the previous one by exactly one blank.

c. The operations permitted in LFM and the associated control cards are as follows:

(1) FILEn The file card identifies the entries which follow until another file card or an execute card as being associated with a single file. FILE1 identifies a LIST mode master file or a LOOKUP mode source file. FILE2 specifies the name of the file to be updated (object file). The word SOURCE may replace FILE1 at the option of the user.

(2) LOCATE This card locates a field in a file. LOCATE cards are essential to define fields when source files are input in card form. They may also be used to define subfields in existing FFS files or to form groups of consecutive fields or subfields. Thus the LFM user is not restricted to the fields and field mnemonics defined in a file's FFT, since he may define his own fields if necessary. Through the use of this operation, the sort key or any part thereof may be accessed by LFM.

(3) DEFINE The DEFINE card may be used to define a literal which is to be called for by name in subsequent control cards Literals may also be defined as they are used, provided they are in the operand section of a card. Literals which are modified during LFM execution must be defined by use of a DEFINE card, since they must be called for by name.

(4) EQUATE The EQUATE card is used to name subfields of FFT - defined fields (EQUATE BEGIN TO ORGIN), and to rename a field to avoid ambiguity when fields from two files are known by the same mnemonic. The format is "EQUATE" (user's made up name) TO (actual field mnemonic).

(5) EXECUTE Identifies the end of the file description section of the control card deck and signals the beginning of the execution portion of the deck.

(6) LIST Specifies that list mode processing is to be done and provides LFM with the mnemonics of the fields upon which the source and object files are to be matched.

(7) LOOKUP Identifies the job as lookup mode and specifies the fields to become search and table arguments during execution.

(8) IF The IF statement is used to condition a subsequent action (such as a move). The IF statement must be satisfied for the subsequent action to take place.

(9) AND The AND statement is also used to condition a subsequent action and must be satisfied for that action to be performed.

(10) OR The OR statement specifies that either this statement or the preceding logic statement must be satisfied for the subsequent action statement to be executed.

(11) MOVE Specifies that one field is to be moved to another. Legitimate move operations are: (1) literal to literal; (2) literal to data file; (3) data file to literal; (4) source file to literal; and (5) source file to object file. Moves of literals, object file fields, and list mode source file fields are stopped by the shortest field involved (e.g., only the final four characters of a six character could be moved to a four-character data file field). Moves of lookup mode source file fields are in most instances controlled by the size of the field into which the data is moved. All moves are made from right to left, beginning with the right-most positions of the respective fields.

(12) STRIP The STRIP instruction causes the non-numeric portion (zone bits) of a single character to be moved to another field. Its primary use is to blank out zone bits in a data field.

(13) DELETE Causes the deletion (blanking out) of the field specified.

(14) DELETE REC Instruction used to completely delete a data record.

(15) * The * card is used to enter comments into the LFM deck and to terminate a logic grouping. Since the * card resets the hit indicator, the analyst must not use it for comments in the midst of a logic grouping.

(16) UPDATE Specifies that the management change data field (CHGDT) should be changed to the run date.

(17) END Signifies the end of the control deck. This card is unnecessary following an LMJOB ID EXECUTE card.

NOTE: The file descriptor cards LOCATE, DEFINE, and EQUATE must follow the FILE n card specifying the file to which they refer. For example:

```
FILE1   XXXXA
EQUATE  FELDA TO FLDID
DEFINE  FELDB @0016@
LOCATE  FELDC 14,21
FILE2  YYYYA
EQUATE  FLD TO FLDID
DEFINE  FELDE @LITERAL TWO@
DEFINE  FLF @THIRD LITERAL@
```

OPCODE	A-FIELD	RELATION	B-FIELD
FILE n 1 for source 2 for object	CARDS - Card input XXXXA - FFS data file XXXXR - FFS answer tape		
SOURCE (optional replacement for FILE1 card)	CARDS - card input XXXXA - FFS data file XXXXR - FFS answer tape		
LOCATE	Mnemonic (1 to 5 characters). The mnemonic must differ from those used in FFTs associated with the job.		Location of the high and low order positions of the field, separated by a comma. Card columns should be used for card input. For FFS data files and answer tapes, consider the first character of the record character count as position one.
DEFINE	Literal Mnemonic (1 to 5 characters)		The literal, enclosed by at-signs.
EQUATE	Mnemonic (1 to 5 characters) made up by user.	TO	DATA file mnemonic and any adjustment if necessary. Adjustment is in the Form "-2" or "-Ø2". The minus must immediately follow the field mnemonic. A plus (+) may not be used.
EXECUTE			
LIST	Master file match field	VS	Data file match field.
LOOKUP	Source file match field (table argument)	VS	Data file match field (search argument).

OPCODE	A-FIELD	RELATION	B-FIELD
IF AND OR	Mnemonic (literal or object file) NB: Any literal used here must exactly match the data field in length. For exampl the operation "IF @WARSAW@ EQ CITY" when CITY in the data record was "WARSAWbbb" would produce a not equal compare and therefore no action.	EQ=equals NE=not equal HI=higher LO=lower HE=higher or equal LE=lower or equal	Mnemonic (literal or object file field) See note under A-FIELD.
MOVE	Mnemonic (literal, source file, or object file field).	TO	Mnemonic (literal or object file field)
STRIP	Mnemonic (literal, source file, or object file field).	TO	Mnemonic (literal or object file field)
DELETE	Mnemonic (object file field)		
DELETE REC			
*			
UPDATE			
END			

4-21. TYPES OF LFM RUNS:

a. Provision has been made for two types of LFM jobs. The first type is the one-time-only run, usually necessary to correct errors, purge records, or change codes. The second type is the recurring run, such as case four in the examples to be described later, wherein skeleton records are completed by LFM on a regular basis.

b. For one-time-only jobs, the console operator should be instructed not to save the new tape produced during the LINKLOADRE phase of the run. For recurring jobs, the operator should be instructed to save and label the new library tape. (For detailed run procedures, see operator's manual.)

4-22. MODES OF OPERATION:

a. Three modes of operation are available to logically maintain a file; direct, list, and lookup. In the direct mode, which may be used in conjunction with either of the other two modes, maintenance is performed based upon the use of externally assigned literals. For example, change a field (FLDAA) to blanks if another field (FLDBB) is equal to a literal (X). This could be stated in LFM language as long as the mnemonics FLDAA and FLDBB were present in the FFT for the file as follows:

```
DEFINE LITLX @X@

IF FLDBB EQ LITLX

DELETE FLDAA
```

b. The direct mode is a straightforward method of performing logical maintenance, and by itself is utilized primarily on a one-time-only basis. On the other hand, the other two modes, list and lookup, are such that their use might be recurring and a standard procedure for certain files.

c. The first of these, the list mode, is designed to perform selective updating of one FFS file by use of a second FFS file. A likely use for this feature is in the use of the AIRØA file to update the CMFLA file. Consider, for the purpose of this example, that the record ID of the AIRØA file uses the field CITYØ for the first field. Consider that the CMFLA file uses ORIGIN as the name of the first field. Also assume that the field OCOOR in CMFLA is a fixed field and that it will be updated by the field COORD of the AIRØA file. The prerequisite for list mode processing is that both the master file (that which supplies data - AIRØA here) and the subordinate file (that to be maintained - CMFLA here) be in the same sort. The record controls of both files do not have to contain the same fields; however, the high-order field (or fields) must be the same. Here the record control of each file has been redesigned temporarily to fulfill this requirement, so when either AIRØA or CMFLA is sorted on the order of the other, selective updating may be accomplished. The fields specified in the LIST card must be the high order record control fields used.

d. To perform this type of update, a record from the subordinate CMFLA file is read and then records from the master AIRØA file are read until a city match is made. The subordinate file record is then updated (if all logic conditions are satisfied) and another CMFLA record read. This record might contain the same city name as the preceding record and therefore be updated by the same master file record. If this is not the case, master file records are read until a city match is made, and maintenance continues.

e. The list mode logical maintenance run illustrated above might be stated as follows:

```
LIST CITY VS ORGIN

MOVE COORD TO OCOOR
```

The fields CITYØ and COORD are defined in the AIRØA FFT, while ORGIN and OCOOR are defined in the CMFLA FFT. (The "Ø" after CITY is automatically added by LFM.)

f. If the fields in CMFLA file had the same names as those in AIRØA file, it would be necessary to redefine the names in one for the LFM run by the use of EQUATE control cards. In the example above the EQUATE entries might be:

```
EQUATE OCITY TO CITY
EQUATE OCORO TO COORD
```

g. In lookup mode processing, maintenance is effected by using a fixed field in the file to be updated as the search argument in a table lookup against tables (generated by LFM from the master file) containing values to be inserted into the subordinate file. This mode is akin to list mode processing, except that (1) record control fields common to the two files are not required, and (2) the two files need not be in the same sort.

h. For example, consider that in coding data for the CMFLA file the coder only filled in the city name and did not list the coordinates. A table listing city names and corresponding coordinates might be used to list those coordinates and save the coder looking them up. This also would save input errors because once a set of coordinates was checked for a city then the chance of error would be reduced by allowing the computer to furnish them. Here the record is first created in skeleton form and later completed by table lookup in a LFM run.

i. To perform lookup mode maintenance the following entries might be made:

```
LOOKUP CITY VS ORGIN
MOVE COORD TO OCOOR
```

Here the contents of CITYØ would be checked against the contents of ORGIN and when a "hit" was made the field OCOOR would be updated with the contents of COORD.

j. Example One - The File in Error. This example represents one of the most common uses of logical file maintenance. File users have discovered a series of errors in the file; errors which are present in several records, yet of-a-kind. In a particular file, a coder has permitted dates to enter his file with subfields reversed, that is, in day, month, year order. This mistake occurred one day only, but the incorrect records are scattered throughout the file. To correct this error using normal maintenance procedures would require that the analyst specify the record control of every record to be corrected, and to specify the correct value once for each record control. Using logical maintenance he must specify only the error and recovery conditions, irrespective of record control. The control cards required for the run might be:

```
LMJOB AA COMPILE
FILE2 CMFLA
DEFINE FIELD @650601@
EXECUTE
IF DATE EQ @010665@
MOVE FIELD TO DATE
END
```

Example One used the assumption that DATE0 was defined as a group consisting of the fields DYEAR, DMNTH, and DDATE.

k. Example Two - Parameter Changes.

(1) In this example, data in a file is no longer valid because of such developments as changes in codes. Once again, numerous records in the file are affected and normal update would involve tedious analyst effort. The changes required follow a logical pattern.

(2) All records which described dead-head and charter flights (i.e., those which formerly had the code letters D and T, respectively, in the field FLTYP) must now replace the code with the letter X. In addition records which presently include an X are to be deleted since we want the new file to contain only passenger, cargo, dead-head and charter flights, with the latter two being designated as "other." The following cards will accomplish the job:

```
LMJOB BB COMPILE
FILE2 CMFLA
DEFINE CROSS @X@
EXECUTE
```

```
IF FLTYP EQ CROSS
DELETE REC
IF FLTYP EQ @D@
OR FLTYP EQ @T@
MOVE CROSS TO FLTYP
END
```

In this example, the order of the statements is critical. The delete record statements must be made before the move statements, for LFM proceeds statement by statement, from front to back through the input deck. If cards four and five were placed in the deck following card eight, not only would records presently containing an "X" in the FLTYP field be deleted, but those containing "D" and "T" as well.

1. Example Three - "Automatic" Update.

(1) There often exists a file which contains information of the utmost validity -- a "master" file. Other files may contain some of the same fields as the "master," but because of lack of update procedures or the complexity of update, these fields may contain outdated information. A method of regular updating of these fields from the "correct" data in the "master" file is required, and this may often be accomplished by the use of logical file maintenance.

(2) But suppose that a new geodetic survey of airports is performed and that the runway coordinates of several airports are changed. After the AIRØA file is updated, CMFLA data fields must also be changed. After the AIRØA file is updated, CMFLA data fields must also be changed. Assuming that ORGIN is in the same sort as CITYØ and that OCOOR is a fixed field ir CMFLA, logical maintenance can update CMFLA by use of the LIST mode. Records from CMFLA (the subordinate file) are matched against records from AIRØA (the master file). Whenever ORGIN equals CITYØ, the contents of the field COORD will be moved into the OCOOR field of CMFLA. The following card deck might be used to accomplish this update:

```
LMJOB CC COMPILE
FILE1 AIRØA
FILE2 CMFLA
EXECUTE
LIST CITY VS ORGIN
MOVE COORD TO OCOOR
END
```

(3) If more than one field (say FELDB, FELDD, and FELDF) in the subordinate file were to be changed to the contents in fields FELDA, FELDC, and FELDE in the master file, the sequence of move cards would have the following format:

```
MOVE FELDA TO FELDB

MOVE FELDC TO FELDD

MOVE FELDE TO FELDF
```

m. Example Four - Reducing Input.

(1) In some cases, logical maintenance capabilities can be used to reduce the amount of analyst-prepared data necessary to create a record. A file containing information of a repetitive nature may be receptive to this work-saving technique. Skeleton records are first built during a standard maintenance run, and repetitive data is added in a subsequent LFM run.

(2) Using the assumption again, the OCOOR is a fixed field, CMFLA might be built without any data in OCOOR. Since there are relatively few airports and the COORD field in the AIRØA file is very seldom changed, data from AIRØA can be used to fill out skeleton records in CMFLA. Even though there is no assumption as to the sort of ORGIN, logical maintenance can still handle this file update using the lookup mode.

(3) In lookup mode, ORGIN is compared to every value of CITYØ using a table lookup procedure (the table is formed by LFM from the AIRØA file). If there is a hit in the lookup, then the move(s) based on this condition are performed. When the lookup fails, no fields are changed. Such a run might be accomplished by use of these LFM entries:

```
LMJOB DD CREATE

FILE1 AIRØA

FILE2 CMFLA

EXECUTE

LOOKUP CITY VS ORGIN

MOVE COORD TO OCOOR

END
```

SAMPLE LOGICAL FILE MAINTENANCE RUN

```
Col. # 6          16        21
       MON$$      JOB       LM
       MON$$      ASGN      MRA,B5
       MON$$      ASGN      MRB,B3,B4,B8 **
       MON$$      ASGN      MRC,A1,A2,A6 **
       MON$$      ASGN      MRD,A3,A4,A7 **
       MON$$      ASGN      MRE,B1,B2,B9 **
       MON$$      ASGN      MRF,B7
       MON$$      ASGN      MRG,A5
       MON$$      ASGN      MWA,FA          --> standard FFS assignment deck
       MON$$      ASGN      MWB,FB *
       MON$$      ASGN      MWC,FC *
       MON$$      ASGN      MWD,FD
       MON$$      ASGN      MWE,FE *
       MON$$      ASGN      MWF,FF *
       MON$$      ASGN      MWG,FG *
       MON$$      ASGN      MWH,FH
       MON$$      ASGN      MWJ,FJ *
       MON$$      ASGN      MWK,FK *
       MON$$      ASGN      LIB,[      --> FFS disk library area.
```

```
Col. # 1
       LMJOB XY COMPILE
       FILE2 CMFLA
       DEFINE FLDAA @UNITEDbbbb@
       EXECUTE                          --> causes execution program to be
       IF ANAME EQ @BRANIFFbbb@             generated. The program ID will
       MOVE FLDAA TO ANAME                  be LXY1P
       END
ol. #  6          16
       MON$$       EXEQ AUTOCODER,,MRE  --> Assembles execution program

       MON$$       EXEQ LINKLOADRE      --> Updates subroutine library
        (OBJECT DECK PRODUCED BY            with execution program
             AUTOCODER RUN)
```

These areas need not be assigned if cross index is not used.
Needed if 3-way merge is going to be used.

```
Col. #  6          16          21
        MON$$                  JOB LM
        MON$$      ASGN        MRA,B5
        MON$$      ASGN        MRB,B3,B4,B8 **
        MON$$      ASGN        MRC,A1,A2,A6 **
        MON$$      ASGN        MRD,A3,A4,A7 **
        MON$$      ASGN        MRE,B1,B2,B9 **
        MON$$      ASGN        MRF,B7
        MON$$      ASGN        MRG,A5
        MON$$      ASGN        MWA,FA
        MON$$      ASGN        MWB,FB *
        MON$$      ASGN        MWC,FC *
        MON$$      ASGN        MWD,FD
        MON$$      ASGN        MWE,FE *
        MON$$      ASGN        MWF,FF *
        MON$$      ASGN        MWG,FG *
        MON$$      ASGN        MWH,FH
        MON$$      ASGN        MWJ,FJ *
        MON$$      ASGN        MWK,FK *
```

→ standard FFS assignment deck

```
        MON$$      EXEQ        LM
```

→ calls in logical file maintenance

```
Col. #  1
        LMJOB XY EXECUTE
```

→ causes execution of generated program LXY1P, i.e., actual update of file

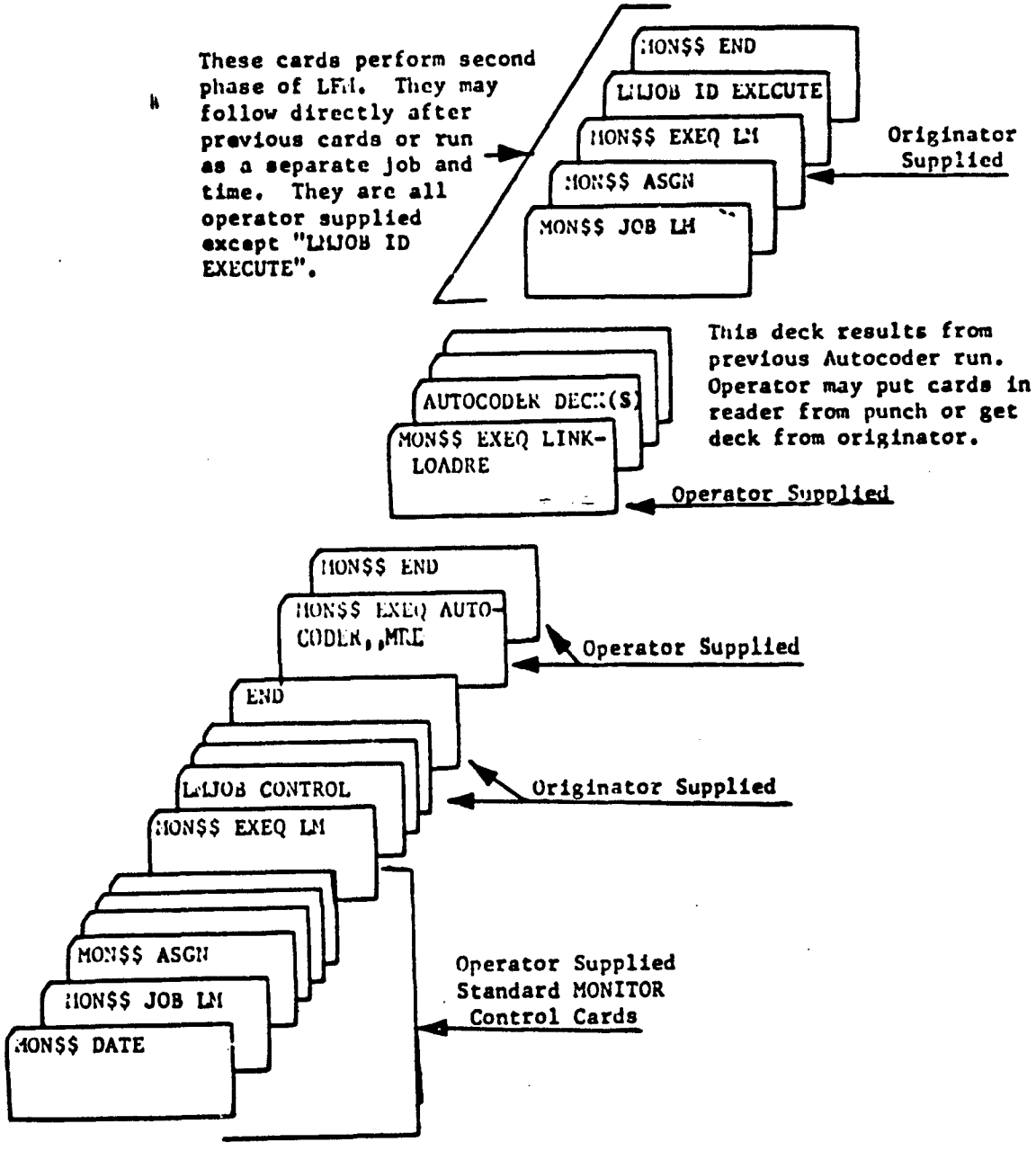
```
Col. #  6          16
        MON$$      END
```

→ terminates run

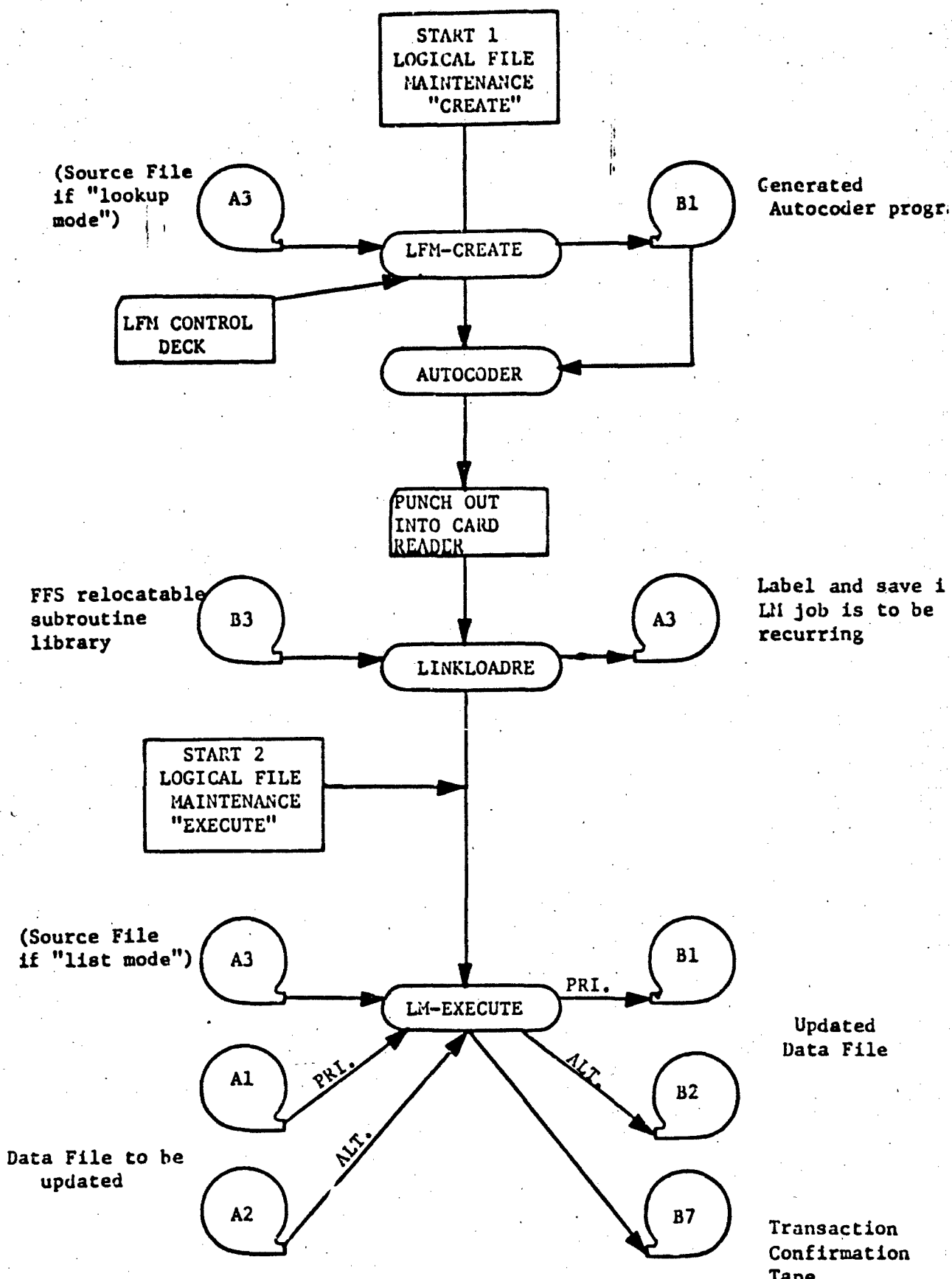
* These areas need not be assigned if cross index is not used.

** Needed if 3-way merge is going to be used.

LFM RUN DECK


These cards perform second phase of LFM. They may follow directly after previous cards or run as a separate job and time. They are all operator supplied except "LMJOB ID EXECUTE".
MON$$ END
LMJOB ID EXECUTE
MON$$ EXEQ LM
MON$$ ASGN
MON$$ JOB LM
Originator Supplied
AUTOCODER DECK(S)
MON$$ EXEQ LINK-LOADRE
This deck results from previous Autocoder run. Operator may put cards in reader from punch or get deck from originator.
Operator Supplied
MON$$ END
MON$$ EXEQ AUTO-CODER,,MRE
Operator Supplied
END
LMJOB CONTROL
MON$$ EXEQ LM
Originator Supplied
MON$$ ASGN
MON$$ JOB LM
MON$$ DATE
Operator Supplied
Standard MONITOR
Control Cards

INPUT-OUTPUT DATA FLOW



LFM 1403 PRINTER MESSAGES

DASH OR COMMA MISSING	(Check the preceding LOCATE card for a missing dash or comma.)
ENTRY EXCEEDS ALLOTTED NUMBER OF _____.	(An entry has exceeded the limits of a specified table.)
FIELD ADDRESS ERROR	(Check field ID to determine if legal field name was used or if ID was properly equated.)
FIELD NOT IN FFT	(Check field ID to determine if it is a legal name in the FFT.)
ILLEGAL BLANK FIELD	(Check to determine if a required field is missing on a control card.)
ILLEGAL CARD OR SEQUENCE ERROR	(Check control deck for proper order or illegal card.)
ILLEGAL OPERATION	(Check the preceding control card to determine if an illegal operation code was used.)
ILLEGITIMATE OPCODE	(Check the preceding control card to determine if an illegal operation code was used.)
LITERAL EXCEEDS 30 CHARACTERS	(A literal in one of the control cards has exceeded the maximum of 30 characters.
LOGIC OPERATOR	(Check preceding control card for proper use of a logical description.
NO BLANK FOLLOWING LAST AT-SIGN	(Check all control cards containing literals to determine if a blank follows the last AT-SIGN.)

NO FFT, FILE XXXXX	(No FFT can be located for the SOURCE file used.)
NO FFT ON SYSTEM	(No FFT can be located for the OBJECT file use.)
NO INPUT TYPE SPECIFIED	(Check the SOURCE or FILE card(s) for the proper description.)
NO LEADING AT-SIGN	(Check control card(s) containing literals for a leading AT-SIGN.)
NO LEGITIMATE OPCODE	(Check the preceding control card to determine if an illegal operation code was used.)
NO TERMINATING AT-SIGN	(Check control card(s) containing literals for a terminating AT-SIGN.)

APPENDIX A

GLOSSARY OF TERMS

ALPHA	(See ALPHAMERIC)
ALPHABETIC	Consisting of alphabetic characters only.
ALPHAMERIC	Consisting of alphabetic and/or numeric and/or special characters.
ALPHANUMERIC	(See ALPHAMERIC)
BLOCK	(1) A physical tape record (separated from other records by interrecord gaps) which contains multiple-logical data records. See "Blocking of Records." (2) A group of computer "words" considered as a unit by virtue of their being stored in successive storage locations.
BLOCK COUNT (Field)	A field which is the first four characters of each block of logical records, containing the number of characters in the block. Do not confuse with "record character count."
BLOCKING OF RECORDS	The combining of multiple-logical records into one block of information on tape, to minimize the time wasted due to acceleration and deceleration of tape, and to conserve space on tape.
CHARACTER	(1) A numeric digit, an alphabetic letter, or a special symbol. (2) The representation of any of the above in (1), in a computer or a storage medium.
CONTROL FIELD	(See RECORD CONTROL FIELD)
CONTROL GROUP	(See RECORD CONTROL GROUP or RECORD ID)
CROSS INDEX	A list of record ID's maintained by unique content of a specified field. This list may be used by retrieval to minimize file search time. A cross index exists only for an indexed file.
DATA	One or more items of information that can be processed by a data processing system.

DATA ELEMENT	A specific item of information appearing in a set of data. As used in this manual, "element" means "field/group/subset."
DATA FILE	Also called "FFS data file" or "formatted file" or just "file." An organized file of related formatted data records called "file records." Since each of the file records is formatted, the file is called a "formatted file." The sequence of file records in the data file is determined by the record ID of each file record. No more than one data file can be on one physical reel of magnetic type. However, multiple reels may be required to store one data file (multireel file).
DATA RECORD	(1) As a general term means a group of related fields of data treated as a unit. (2) Synonymous with "FFS file record" (see FILE RECORD).
EDIT	The modification of data by using the editing capabilities of the 1410 DPS to: Insert characters before, after, or between characters of data. Suppress leading zeros. Selective inserti n of the credit symbol, minus sign, asterisk, dollar sign, and decimal.
ELEMENT	(See DATA ELEMENT)
EXTERNAL FORMAT (E.F)	A format for the FM input file in which data elements occur in definite positions within particular input record types; as described by an input descriptor deck (IDD). Multiple-data elements may be contained in one external format input record.
FFS DATA FILE	(See DATA FILE)
FFS DATA RECORD	(See FILE RECORD)
FFS RELOCATABLE EXECUTION LIBRARY	A file of relocatable conversion subroutines/ tables, FFT's, and RIT's, which are used by the various FFS programs.
FFS RELOCATABLE PROGRAM LIBRARY	A file of relocatable programs comprising the Formatted File System. It must replace or be part of the System Library before it can be used.
FIELD	The smallest defined logical unit handled by the Formatted File System, consisting of one or more adjacent data characters.

FILE	Generally a nonspecific term meaning an organized collection of information directed toward some purpose. However, in this manual "file" means "FFS data file," unless otherwise qualified. (See DATA FILE.)
FILE FORMAT TABLE (FFT)	A table consisting of 9 logical records which completely describes an FFS data file. This table is generated by file structuring. There is one FFT for each data file. The FFT's are kept on the FFS Relocatable Execution Library.
FILE ID	Name of the Formatted File System data file.
FILE MNEMONIC	Same as FILE ID.
FILE RECORD	(Also called data record.) A logical record of related fields of data. The file record is formatted, that is each element of the file record has been defined, identified, and assigned a relative position in the file record. Each file record has a fixed set which contains the record ID. The file record may also contain a number of periodic sets and a variable set. File records are the largest components of an FFS data file.
FILE SYNONYM TABLE	Each Formatted File System data file may have one synonym table, which allows the retrieval technician to use more meaningful element names in place of the five character mnenomics in the File Format Table (FFT).
FIXED FIELD	A field defined in the fixed set of a file record and which must appear once and only once in the file record.
FIXED GROUP	(See GROUP)
FIXED SET	That portion of a file record consisting of all the fixed fields/groups of the file record, including the record character count, periodic set control, and variable set control fields.
FORMAT	A predetermined arrangement of characters, fields, or other data. A format does not describe the data, but describes its organization.

FORMATTED FILE	(See DATA FILE)
GROUP	A collection of one or more adjacent fields of the same set that are related. A group is capable of being processed or otherwise manipulated as a unit. The system treats a group the same as a field. The fields within a group in no way lose their individual identities, and may be treated as if they were not grouped. If fixed fields are grouped, the group is a "fixed group." A "periodic group" is a grouping of periodic fields.
HIGH ORDER POSITION (HOP)	The left most (most significant) position of a field.
IBM RELOCATABLE LIBRARY	(See SYSTEM LIBRARY)
INPUT DESCRIPTOR DECK (IDD)	A deck of cards which describes the external format input file, and which must precede each external format input file.
INPUT FILE	A card or tape file which contains all or a portion of the data needed by file maintenance to update an FFS data file. If the input file is internal format, it must be cards or card images on tape. If the input file is external format, it may be cards, card images on tape, or noncard image tape.
INPUT GROUP	All of those input records containing information to be extracted for the purposes of creating or updating a single (the same) file record.
INPUT GROUP CONTROL FIELD	Either an artificial control field or an actual data field (or fields) by which the input file is sorted or manually arranged prior to input to the system. This is done so that all input records belonging to the same input group (i.e., pertaining to the same file record) will be grouped together in the input file.
INPUT RECORD	A single card or tape record in an input file.
INPUT RECORD TYPE CODE	The code in the input records used to distinguish one input record type from another.

INTERNAL FORMAT (I.F.)	A format for the FM input file in which a single data element occurs per input record. The description of internal format input records does not require an IDD, since each record follows a predefined format and contains descriptive data such as record ID, element name, and operation code.
LOGICAL RECORD (LR)	A record (usually on tape), which is terminated (logically) with a record mark ‡, but is not necessarily begun or terminated by an inter-record gap, as is a physical tape record. One physical tape record ("block") may contain many logical records.
LOW ORDER POSITION (LOP)	The right most (least significant) position of a field.
MERGE	To combine items into one sequenced file from two or more similarly sequenced files, without changing the order of the items.
MNEMONIC	Usually a fixed length abbreviation of a name, which assists human memory of the name. An example is CMFLA for Commercial Flights File.
MULTIREEL FILE	A file so large as to require more than one physical reel of tape for storage.
PAD	To add characters to a data element (word, field, group, record, etc.) in order to increase its size to a predetermined, fixed value.
PERIODIC FIELD	A field defined in a periodic set of a file record which may appear more than once in a file record.
PERIODIC GROUP	(See GROUP)
PERIODIC SET	A collection of 1 to 599 periodic subsets having the same formats.
PERIODIC SUBSET	One or more uniquely defined periodic fields/groups, which are so related that if repeated, must be repeated as a unit.
RECORD	A grouping of facts or fields of information treated as a unit. A logical data group.

RECORD CHARACTER COUNT (FIELD)	A field which is the first four characters of every FFS file record, and as such is the first fixed field of each file record. It contains the count of characters in the file record.
RECORD CONTROL FIELD	One of the fixed fields that comprises the record ID (record control group).
RECORD CONTROL GROUP	(See RECORD ID)
RECORD ID (Also called "record control group" or "control group.")	The initial data field(s) of the fixed set which make each file record in a file unique, and are used to identify the file record. The file records in a file are sequenced according to the contents of their "record control group" or "record ID."
SORT	To arrange items or information into sequence, based upon the data content of some field (sort key) contained within the items.
SUBROUTINE	As used in the FFS, means the set of instructions (subprogram) in relocatable form which are used to perform data conversions for input, output, and retrievel purposes. These subroutines are kept on the FFS Relocatable Execution Library, and are used by the various FFS programs.
SUBSET	(See PERIODIC SUBSET)
SYNONYM TABLE	(See FILE SYNONYM TABLE)
SYSTEM LIBRARY	(Refers to the Operating System Library.) A file of relocatable programs created or supplied for use under control of the system monitor.
SYSTEM MONITOR	The system monitor controls the 1410/7010 Operating System. Some of the functions it performs are: (1) Assignment of input/output units. (2) Program loading including relocation of programs, and linkage between programs and subroutines that were independently written and compiled. (3) Program transition from run to run and job to job, and communication with the operator. (4) Sequencing and monitoring of the programs of the operating system and the FFS programs.

SYSTEM OPERATING FILE (SOF)	The file (disk or tape) which contains (in absolute form) the: (1) 1410/7010 Operating System (including system library). (2) All of the programs comprising the 1410 FFS.
TABLE	A conversion subroutine of the "table look-up" type. See SUBROUTINE.
TRUNCATE	To drop one or more characters from a data element without altering the remaining characters. For example, to truncate 4.149 on the right by one character would result in 4.14 instead of 4.15.
VARIABLE FIELD	Identical to the more common term "variable set," since the variable set consists entirely of the variable field.
VARIABLE SET	A single, unformatted, variable length, non-repetitive field which may occur immediately after the last periodic set in a file record.
1403	Used to refer to the high-speed line printer on the 1410 DPS.
1415	Used to refer to the console typewriter of the 1410 DPS.

APPENDIX B

ABBREVIATIONS USED

Alpha	Alphameric
¥	Often used in this manual to indicate a blank position (no punches and/or no printing).
CAD	Create, Change, or ADD
Char.	Character
CHG	Change
CHGDT	Mnemonic for the program maintained (fixed set) field meaning the date the file record was last changed.
CMA	Change or ADD
Col.	Column
CRE	Create
CREDT	Mnemonic for the program maintained (fixed set) field meaning date the file record was created.
Ctl. or Ctrl.	Control
DATE	File structuring option to insert Create and Change data fields into the fixed set of the FFT.
DEL	Delete
DPS	Data Processing System
EAM	Electrical Accounting Machine
E.F.	External Format
FFS	Formatted File System
FFT	File Format Table
FG	File Generation
Fld.	Field

FM	File Maintenance
FR	File Revision
FS	File Structuring
Grp.	Group
HOP	High Order Position
ICC	Input Control Card
ID	Identity or Identification (i.e., name or mnemonic).
IDD	Input Descriptor Deck
I.F.	Internal Format
IOCS	Input-Output Control System
IP	Input Processor
LFM	Logical File Maintenance
LM	Logical Maintenance
LOP	Low Order Position
LSTX	File structuring option to obtain up to 9 extra listings on the 1403 printer of the FFT.
LR	Logical Record
n	Number (as in PSSQn)
NEX	No Exchange
NODK	File structuring option to inhibit punching the FFT deck.
Numer	Numeric
OP	Output Processing
Opcode	Operation Code
PCAM	Punched Card Accounting Machine

PSCTn	Periodic Set Control Number
PSSQn	Periodic Subset Sequence Number
RECCT	Record Character Count
RIT	Report Instruction Table
RT	Retrieval
Seq.	Sequence
SFT	Source Format Table
SIU	System Input Unit
SOF	System Operating File
SOP	System Output Program (Output Processor Program)
Specs.	Specifications
VSCTL	Variable Set Control Field
VSET	Variable Set
#	Number (as in LR#2).

APPENDIX C

GENERAL DESCRIPTION OF THE 1410/7010 OPERATING SYSTEM

1. PURPOSE OF THE 1410/7010 OPERATING SYSTEM: The 1410/7010 Operating System is an integrated set of programs and programming systems that provides a 1410 or 7010 installation with a convenient and efficient means of fulfilling its data processing requirements. The fundamental purpose of the operating system is to enable the writing, assembling, and execution of programs with a minimun of programmer time, machine time, and machine-operator time. Time savings are achieved by means of programming systems, service programs, and the system monitor. Each of these is discussed below in general terms. Somewhat more detailed descriptions of the various components of the 1410/7010 Operating System are provided later in this appendix.

a. System Monitor.

(1) The system monitor is the 1410/7010 supervisory program that calls in prespecified programs or routines into main storage as required. In accordance with control information supplied by the user, system monitor provides compile-and-go and batch processing capabilities. The basic features provided by the system monitor are maximum utilization of machine time and improved communication between programming and machine room personnel. Through its compile-and-go and batch processing capabilities, the system monitor substantially reduces the inoperative time between runs. By providing simplified and standard operating procedures to be followed, communication between programming and machine room personnel is greatly improved.

(2) The system monitor works in conjunction with several other elements of the 1410/7010 Operating System. A number of those elements (such as IOCS and linkage loader) are contained within the structure of the system monitor, thereby eliminating the need for the user to include those elements in his own program.

(3) More details on the structure and features of the system monitor are included in this appendix under the heading "Using the Operating System."

b. Programming Systems.

(1) A programming system consists of a language and its associated processor. The programming systems available with the 1410/7010 Operating System are Autocoder, COBOL, and FORTRAN.

(2) Use of the Autocoder, COBOL, or FORTRAN languages reduces the time required by programmers to write and debug programs. The writing of source programs is simplified and thereby speeded up solely as the result of the symbolic nature of these languages. The ability to use a single source language instruction or statement to produce several machine language instructions further reduces programming time.

(3) A great deal of debugging time is saved because many groups of instructions in the object program are generated by the language processors,

and have been pretestested and debugged. These groups of instructions result from Autocoder macro-instructions, and from COBOL and FORTRAN statements.

(4) If an installation decides that their data processing application does not require one or more of the programming systems, the installation may exclude the components they do not desire from their operating system.

c. Service Programs.

(1) The service programs provided in the 1410/7010 Operating System are collections of prewritten routines that perform specific functions in accordance with control information supplied by the user. Some programs in this category function independently, while others are designed for use in conjunction with a user's program. The service programs include the tape sorting program, the input/output control system, the system generation program, the tele-processing supervisor, and the four 1410/7010 utility programs: "snapshot," "storage print," "tape print," and "1301 print."

(2) Use of the service programs to perform various functions required by an installation conserves many hours of valuable programming time. Each service program is designed to perform its function efficiently, making maximum use of an installation's specific machine configuration. Each routine of every program has been thoroughly pretested.

2. THE AUTOCODER PROGRAMMING SYSTEM: The Autocoder Programming System provides a convenient and efficient means of writing programs. Features provided in the Autocoder language and processor not only simplify the task of writing source programs, but also facilitate the running and debugging of object programs. Among these features are the following:

Mnemonic Operation Codes
Label Processing
Macro System
Relocatable Object Programs
Assembly Listings

a. Mnemonic Operation Codes. The operation codes in the Autocoder language have a mnemonic relationship to the machine instructions with which they are associated, thereby greatly simplifying the task of the programmer who would otherwise be required to work with the abstract language of the computer. For example, the mnemonic operation code "M" is easier for a programmer to remember and associate with the operation "multiply" than its machine language equivalent "@."

b. Label Processing. Another symbolic feature of Autocoder is the facility for assigning a name or "label" to a specific location or area in core storage, and referring to that label thereafter, rather than to the actual core-storage address. By assigning a label to a data field that is easily associated with the data it contains (such as ITEM, SALARY, DATE) or to a routine that is easily associated with the function performed by that routine (such as PROCESS, ERRORCHECK, etc.), a programmer can impart to his program a structural clarity and ease of reference that is impossible in machine language coding.

c. Macro System. The Autocoder Macro System provides facilities for the creation and processing of macro-instructions. A macro-instruction permits the programmer to specify, in one instruction, a series of related operations to be performed. Macro-instructions are translated by the Autocoder processor into the machine language instructions required to perform the indicated operations. These macro-instructions are contained in a "macro-library," contained within the system operating file. There are two advantages in the use of macro-instructions as opposed to "one-for-one" symbolic instructions. First, macro-instructions permit use of the symbolic features of a language (mnemonic operation codes and label processing) at a higher level. Second, they permit the user to write his program by describing the functions he wants performed rather than by describing the operations necessary to perform those functions. In other words, a macro-instruction is used to specify what function is to be performed, and will result in a series of instructions that specify how that function is to be performed.

d. Relocatable Object Programs. Object Programs produced by the Autocoder processor are in relocatable format. This format offers two advantages. First, it permits the program to be loaded into any available area of core storage during batch processing. Second, it permits independently compiled programs to refer to labels and locations in other programs. These references are then linked when the programs are combined during batch processing.

e. Assembly Listings. Assembly listings are produced for all programs compiled by the Autocoder processor. These listings show the source program relation to the object program. Labels are related to core storage assignments, symbolic instructions to their machine language equivalents, etc. The listing is arranged in a format that permits ease of reference. In addition, coding errors detected by the processor are indicated in the listing, making it a valuable aid in program debugging.

3. THE COBOL PROGRAMMING SYSTEM:

a. The COBOL Programming System enables users to create business oriented object programs with a minimum of programmer effort. COBOL (Common Business Oriented Language) source programs are written in a language similar to ordinary business English, thereby permitting the user to associate his program more directly with the related problem.

b. COBOL provides all the advantages of any symbolic language, such as Autocoder, plus the additional advantages inherent in a "higher-level" language. The majority of words in the vocabulary of a higher-level language such as COBOL are at least the equivalent of an Autocoder macro-instruction with relation to the number of machine language instructions produced. Therefore, if COBOL is appropriate for a particular problem, its use can result in the creation of an object program with a minimum of programmer time and effort.

c. The COBOL processor compiles source language programs directly into relocatable machine language, bypassing the intermediate stage of a lower-level symbolic language program. This "direct translation" method reduces compilation time.

d. Object programs produced by the COBOL processor are in relocatable format. This format permits FORTRAN-compiled and/or Autocoder-compiled subroutines to be incorporated into an independently compiled COBOL program at object time.

4. THE FORTRAN PROGRAMMING SYSTEM:

a. The FORTRAN Programming System enables users to create programs for scientific problems with a minimum of programmer effort. FORTRAN (FORmula TRANslation language) source programs are written in a language containing terminology similar to mathematics, thereby permitting the user to associate his program more directly with the related problem.

b. The FORTRAN Programming System offers a "higher-level" language and provides macro-instruction facilities. The FORTRAN processor compiles source programs directly into relocatable machine language programs.

c. The FORTRAN source programs can be written for use independently or as subprograms, i.e., programs that can be combined for execution as a unit along with other programs. Subprograms can contain references to labels in other subprograms, and these will be linked when the subprograms are combined at object time.

5. INPUT/OUTPUT CONTROL SYSTEM (IOCS):

a. The Input/Output Control System (IOCS) is a set of prewritten routines that performs all of the input/output functions for an object program, as well as the components of the operating system itself. Among these functions are scheduling of read and write operations, error detection, and correction, end-of-file handling, checkpoint and restart facilities, and blocking and deblocking of records.

b. The 1410/7010 IOCS is contained within the structure of the system monitor. IOCS provides macro-instruction facilities for unit record equipment, magnetic tape units, 1301 disk storage, the 1414 input/output synchronizer, and the 7750 programmed transmission control. Routines for disk storage permit both random and sequential processing. Random processing is controlled within IOCS by the random processing scheduler. Routines for tele-processing equipment, i.e.,the 1414 and 7750 work in conjunction with the tele-processing supervisor.

c. When the user defines his machine configuration during system generation, only those IOCS routines he will require become part of the system monitor section of his system operating file. IOCS resides in core storage when programs are being executed.

d. Macro-instructions provided for use with IOCS are of two types: one type results in linkages to routines within IOCS; the other type results in generated (open) routines within the user's programs.

e. Input/output functions required by component programs of the

1410/7010 Operating System are automatically performed by IOCS. Programs written by the user for operation within the framework of the operating system must also use IOCS.

6. GENERALIZED TAPE SORTING PROGRAM:

a. The Generalized Tape Sorting Program is a set of prewritten routines provided in relocatable form on the master file (see "Definition of Terms"). It also contains a "sort definition" program that is similar to a compiler. This program provides the user with the facility for creating a sorting program for specific tape files during system generation or during batch processing. In other words, the user can maintain a file of sorting programs in absolute format, and at the same time retain the facility for creating additional absolute sorting programs at object time by means of the sort definition program on the system operating file. The absolute sorting programs operate in accordance with control information specified by the user.

b. The Generalized Tape Sorting Program operates in conjunction with IOCS and other components of the system monitor. It has the ability to sort or merge data records on tape in ascending or descending sequence. It will handle fixed-length or variable-length records, and can include linkages to special routines written by the user.

7. UTILITY PROGRAMS: Four utility programs are provided with the 1410/7010 Operating System: "snapshot," "storage print," "tape print," and "1301 print." Output from these programs is obtained on the standard print unit. All four programs use a standard scheme of abbreviation for nonprintable characters.

a. Snapshot. The snapshot utility program prints all or selected areas of core storage at intervals specified by the user. The snapshot program can be combined with the user's program by the system monitor. Information will be printed as it appears in core storage, with word marks indicated above the appropriate characters. The settings of various indicators and registers will also be listed.

b. Storage Print. The storage print utility program prints all or selected areas of core storage in accordance with control information provided by the user. Unlike the snapshot program, it will be executed only when specifically requested at object time by user-supplied control cards. The printed listing is provided in machine language. Word marks will be indicated where they appear in core storage. Settings of various indicators and registers will also appear in the printed listing.

c. Tape Print. The tape print utility program prints all or a portion of the contents of a magnetic tape. It may print the contents of one or more files, or a specified number of data records within a file. Records can be fixed-length or variable-length format, and odd or even parity. Printed listings may show word separator characters as separate characters or as word marks above the appropriate characters. An end-of-file message is printed after each complete file. Tape identification, mode, and parity are indicated; data record and character counts are also made.

d. 1301 Print. The 1301 print utility program prints all or selected areas of 1301 disk storage. Data to be printed can be in disk storage with or without wordmarks (move mode or load mode) and will be written out in the full track mode of operation.

8. THE TELE-PROCESSING SUPERVISOR:

a. The tele-processing supervisor is an optional component of the 1410/7010 Operating System available to installations with tele-processing devices. Like the input/output control system, the tele-processing supervisor becomes part of the system monitor at system generation. The organization and size of the tele-processing supervisor is determined by the user's configuration of tele-communications devices.

b. The tele-processing supervisor works in conjunction with IOCS and programs provided by the user to handle all input/output functions for tele-communications devices. Its primary function is to supervise the flow of data received from or scheduled for telecommunications devices and to transfer control, when necessary, to appropriate routines within the system monitor or the user's program(s). Unless telecommunications devices are an integral portion of the data processing job performed under supervision of the operating system, the tele-processing supervisor is not included as a component of the operating system. Thus, the typical 1410 IDHS will not use the tele-processing supervisor.

9. THE RANDOM-PROCESSING SCHEDULER: The random-processing scheduler is an optional component of the 1410/7010 Operating System. It augments the basic input/output control system component of the operating system by providing facilities for the efficient handling of input/output operations in "random processing" applications. By using random-processing scheduler an efficient random processing job may be planned and written with simple, sequential macro-type procedures provided. However, during the execution of the job the random processing scheduler causes the individual procedures to be performed in the most efficient order rather than sequentially. Data to and from individual procedures is buffered via input/output stacking areas and disk operations are overlapped with processing. By always performing the individual procedure that can be most expeditiously executed at a given time, the amount of idle time (when one procedure must wait on another) is minimized. Unless an integral part of the data processing performed within the framework of the operating system is a random-processing application, the random-processing scheduler is not included within the operating system. Thus, the typical 1410 IDHS may well not use the random-processing scheduler.

10. SYSTEM GENERATION:

a. Desired optional and the required components of the 1410/7010 Operating System provided by IBM are combined by the user (along with any special components supplied by the user) to create a specific operating system for his own installation. This process is called system generation. The basic file in the user's operating system is the system generator file (see "Definition of Terms"). One or more system operating files can be created for use within an installation from the system generator file.

b. The 1410/7010 Operating System is made available to users on a reel of tape which becomes the master file for the installation. This master file will be made up for use with a tape-oriented system or a disk-oriented system, whichever is specified. From this tape, the user creates a system generator file for his installation. The system generator file is then used to create specific system operating files in accordance with control information provided by the user. An installation's system operating files can include any relocatable and absolute programs supplied by the user, in addition to the required and optional components of the operating system supplied by IBM.

c. In addition to creating one or more system operating files, system generation can include the creation of a system relocatable library file (see "Definition of Terms"). System generation routines can also be used to update the SGF, to update SOF's and system relocatable libraries, and to print a listing of the elements contained on the generated SOF(s). A detailed listing of the contents of the macro library may also be printed. System Operating files can be created for use on tape or in 1301 disk storage. Figure C-1 depicts the general flow of data during the creation of an operating system. It is not a complete chart, and is shown only to facilitate a basic understanding of system generation.

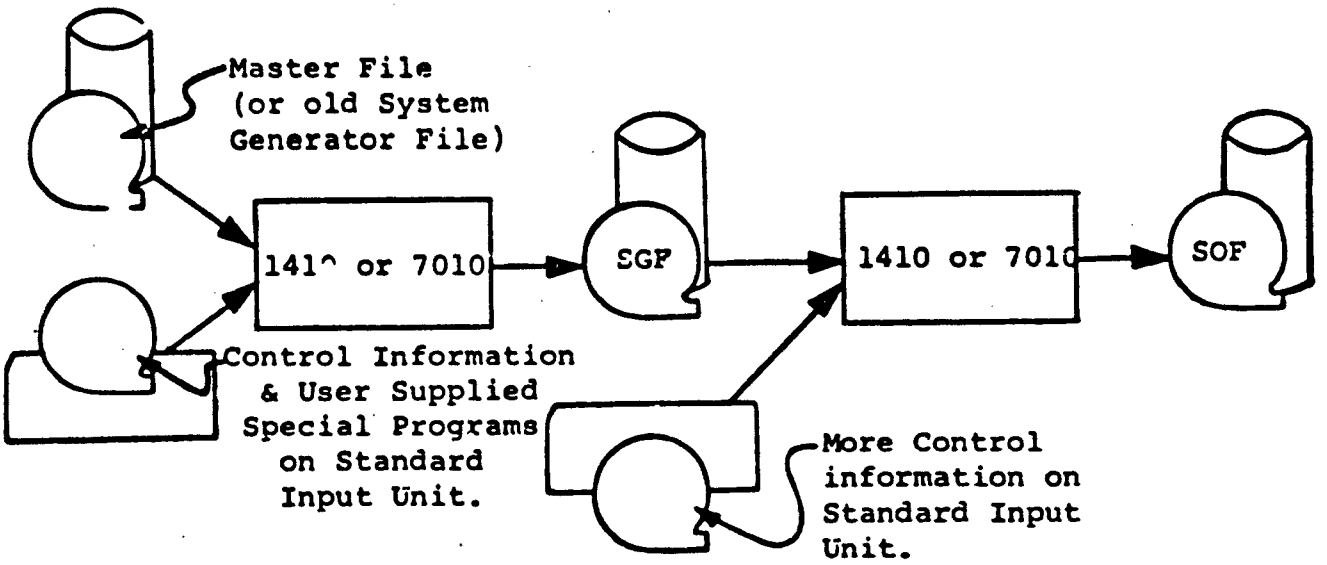


Figure C-1. System Generation, General Data Flow.

11. USING THE OPERATING SYSTEM:

a. Once the operating system has been generated for use by a particular installation, compilations of source programs and the testing and execution of object programs can be performed under control of the system monitor, provided the necessary components of the operating system are available. For example, the COBOL processor must be included on the SOF if a COBOL source program is to be assembled.

b. The system monitor is the heart of an operating system. It performs such major functions as the assignment of input/output units, furnishing transition between jobs, program loading and relocation, and the linkage of independently compiled programs. It also provides batch processing and compile-and-go capabilities.

c. The system monitor contains three major elements: the resident monitor, the transitional monitor, and the linkage loader.

d. The resident monitor consists of control routines that remain in core storage while the operating system is functioning. It includes the operating system's IOCS, input/output assignment routines, end-of-program routines, an absolute program loader, and other frequently used routines. In a tele-processing system, the tele-processing supervisor is part of the resident monitor. The transitional monitor contains routines required to permit transition from run to run and from one job to the next during batch processing. The linkage loader performs the functions required to convert relocatable programs into absolute programs for execution. The transitional monitor and the linkage loader are called into core storage when required.

e. All jobs to be performed in an operating system are controlled by the system monitor in accordance with instructions provided by the user via control information entered via the standard input unit.

12. DEFINITION OF TERMS: The general terms listed below are defined for use within this appendix and other documents concerned with the 1410/7010 Operating System. Terms associated with a particular component of the operating system are defined within the publication describing that component.

a. Absolute Program. A machine language program with actual addresses in a format ready for loading directly into only one specific area of core storage for execution.

b. Alternate Input Unit (AIU). An input unit that can be substituted for the standard input unit (SIU) to permit interruption of batch processing for a high-priority job or for unscheduled diagnostic routines.

c. Batch. A collection of jobs to be performed under the supervision of the system monitor.

d. Batch Processing. The processing of a collection of jobs by the computer without the need for operator intervention between jobs.

e. Compilation. See "Processor."

f. Compile-and-Go. The compilation and subsequent execution of one or more programs within one job without the need for operator intervention.

g. Job. One or more runs specified by the user to be compiled and/or executed as a logical unit without need for operator intervention.

h. Master File. The tape file, provided by IBM, containing all elements of the IBM 1410/7010 Operating System. The master file is the source file for each installation's initial system generation run.

i. Module. See "Subprogram."

j. Processor. A machine language program that translates source programs written in a symbolic language (i.e., Autocoder, COBOL, FORTRAN) into machine language instructions. This is called a "compilation."

k. Relocatable Program. A machine language program or subprogram in a format that allows reassignment of addresses, thus making possible its conversion into an absolute program with addresses adjusted to correspond to any available area of core storage. This format also enables effective communication among several subprograms constituting a single, complete program.

l. Run. A major function performed by a computer, such as the execution of a compiler or an object program, without the need for operator intervention. Also see "Job."

m. Standard Input Unit (SIU). A card reader or magnetic tape unit specified by the user from which control information for the system monitor and other elements of the operating system can be read. It may also serve as the input medium for source program or other data.

n. Standard Print Unit (SPR). A printer or tape unit specified by the user to receive printer output from all programs operating within the framework of the 1410/7010 Operating System.

o. Standard Punch Unit (SPU). A card punch or tape unit specified by the user to receive card punch output from all programs operating within the framework of the 1410/7010 Operating System.

p. Subprogram. The basic program element within the operating system. A subprogram can be a complete program in itself or a program segment (such as a subroutine) that is to be combined with other program segments to form a complete program.

q. System Generator File (SGF). The tape or disk file containing elements of the operating system, selected from the master file, which are needed to satisfy the total processing requirements of a specific installation. This file may also contain user-originated subprograms.

r. System Relocatable Library. A file of compiled relocatable subprograms created for use under control of the system monitor.

s. System Monitor. The supervisory program in the 1410/7010 Operating System that calls prespecified programs or routines into core storage as required. The system monitor, which operates in accordance with control information supplied by the user, provides compile-and-go and batch processing capabilities.

t System Operating File (SOF). A tape or disk file, created by the user from the system generator file, containing absolute programs including the system monitor, which satisfy the particular processing needs of an installation. This file, if on tape, may also contain the system relocatable library.

13. 1410/7010 OPERATING SYSTEM PUBLICATIONS: Users requiring more information about the 1410/7010 Operating System may refer to the following IBM publication

Name	Form Number
Basic Concepts	C28-0318
System Monitor	C28-0319
System Generation	C28-0352
Operator's Guide	C28-0351
Basic IOCS	C28-0322
Utility Programs	C28-0353
Autocoder	C28-0326
Fortran	C28-0328
Cobol	C28-0327
Generalized Tape Sorting Program	C28-0354
Tele-processing Supervisor	C28-0321
Random Processing Scheduler	C28-0323

APPENDIX D

CHARACTER DEFINITION TABLE

The following table defines the ambiguous characters used in this manual.

1 No Print (or blank)

Character Set Used In This Manual	IBM Card Code	BCD Code
$	11-3-8	CB821
¤	12-4-8	CBA84
@	4-8	C84
#	3-8	821
&	12	CBA
%	Ø-4-8	A84
/	Ø-1	CA1
-	11	B
Δ	11-7-8	B8421
¥	No Punches	C

APPENDIX E

Math formula to determine the maximum number of fields/groups allowed for your file under FFS.

Symbols used:

F = Total number of fields/groups in the file. This total includes the following automatically created fields in addition to data fields/groups:

Character Count Field (COUNT)

Create Date Field (CREDT).*

Change Date Field (CHGDT).*

A Periodic Set Control Field (PSCTn) for each Periodic Set.

A Periodic Subset Sequence Field (PSSQn) for each Periodic Set.

A Variable Set Control Field (VSCTL) for a Variable Set.

* - Optional.

S = Total of a fixed set plus the number of periodic sets plus a variable set in the file.

E = Total number of output edit fields required for the file. This total may be assumed to be F/10.

$\bar{A}$ = Average size (number of characters) of the variable portion of a Logical Record 2 entries (Input Parameter Section).

$\bar{T}$ = Average size (number of characters) of the variable portion of a Logical Record 6 entries (Output Parameters and Labels).

$\bar{O}$ = Average size (number of characters) of the Output Edit fields.

N_T = Total character size of the FFT. (Current limitation is 11,000 characters.)

General Equations:

LR	IDENT	FORMULAE
1	ID Record	$N_1 = 15$
2	Detail Record	$N_2 = 5\ (12 + \bar{A})F$
3	Fld/Gp Lookup Tbl	$N_3 = 6 + 13F$

LR	IDENT	FORMULAE
4	Set Table	$N_4 = 6 + 11S$
5	Input Edit	$N_5 = 5$ (assuming no input editing)
6	Extract Table	$N_6 = 5 + (8 + \bar{T})F$
7	Output Edit	$N_7 = 5 + E\bar{O}$
	Total FFT size	$N_T = N_1 + N_2 + N_3 + N_4 + N_5 + N_6 + N_7$

Combining these general equations one can estimate the maximum number of fields/groups that can be placed in file by solving this equation.

$$F = \frac{11000 - E(3+\bar{O}) - 42 - 11S}{33 + \bar{A} + \bar{T}}$$

or assuming $E = \frac{F}{10}$

$$F = \frac{11000 - 42 - 11S}{33 + \bar{A} + \bar{T} + (3+\bar{O})/10}$$

Example 1:

Based on the sizes of particular elements in previously designed FFTs, the following averages were established:

$S = 7 \quad E = \frac{F}{10} \quad \bar{A} = 9 \quad \bar{T} = 14 \quad \bar{O} = 10$

Solving the above equation for F using these assumptions provides the following maximum number of fields that can be contained in an average FFT.

$$F = \frac{11000 - 42 - 77}{33 + 9 + 14 + 1.3}$$

$$F = \frac{10881}{57.3}$$

$$F = 190$$

Example 2:

In order to maximize the number fields/groups one can minimize the size of $\bar{A}$ and $\bar{T}$. For example assume the following:

$$S = 10 \qquad E = \frac{F}{10} \qquad \bar{A} = 4 \qquad \bar{T} = 2 \qquad \bar{O} = 10$$

Solving the above equation:

$$F = \frac{11000-42-110}{33+4+2+1.3}$$

$$F = \frac{10848}{40.3}$$

$$F = 269$$

INDEX

	Page
AND	4-77
ASTERISK PROTECTION	3-14
AUTOMATIC UPDATE	4-84
BLAN	4-69
C.I.U	2-18
CARD SEQUENCE	3-31
CHANGE MODE	2-13
COMPILE	4-76
CONTROL CARD ENTRIES	4-76
CONTROL LOCATION SECTION	4-41
CREA	4-69
CREATE	4-76
CREATE MODE	2-13
CROSS-INDEXING	2-25
CROSS-INDEX UPDATER	2-24
CROSS-INDEX UPDATER PHASE	2-19
CROSS-INDEXED FILES	2-18
DATA ELEMENT PLACEMENT	3-5
DATA FILE	2-1, 2-22
DATA RECORD	2-1, 2-22
DBLA	4-69
DECIMAL CONTROL	3-17
DEFINE	4-77
DELETE	4-78
DELETE REC	4-78
DIRECT	4-81
EDIT CARD	3-18
ELIM CARD	3-28
END	4-78
EQUATE	4-77
ERR	4-69
EXECUTE	4-76, 4-77
EXTERNAL FORMAT	2-16, 4-11
EXTERNAL FORMAT INPUT	4-35
FFS	1-1
FFS LIBRARIAN	1-4
FFT	2-24
FG	2-24
FIELD	2-22
FIELD CARD	3-21
FIELD EXTRACT CARD	4-54
FIELD EXTRACTION SECTION	4-41
FIELDS	2-2
FILE	2-1, 4-77
FILE FORMAT	2-22
FILE FORMAT TABLE	3-34

FILE GENERATION ---------- 1-4, 2-12
FILE GENERATION FG ---------- 2-8
FILE IN ERROR ---------- 4-83
FILE MAINTENANCE ---------- 1-4, 2-13, 4-1
FILE MAINTENANCE FM ---------- 2-8
FILE MAINTENANCE PROPER PHASE ---------- 2-17
FILE MAINTENANCE SUPERVISOR PHASE ---------- 2-15
FILE RECORD ---------- 2-1, 2-2, 2-22
FILE RECORD FORMAT ---------- 2-4
FILE REVISION ---------- 2-13
FILE REVISION SUMMARY ---------- 3-96
FILE STRUCTURING ---------- 2-12
FIXED DATA ---------- 2-22
FIXED DATE ---------- 2-1
FIXED FIELD ---------- 2-22
FIXED FIELDS ---------- 2-2, 4-25
FIXED GROUPS ---------- 2-3, 4-25
FIXED SET ---------- 2-3, 2-6, 2-23
FLOATING DOLLAR SIGN ---------- 3-15
FM PROPER ---------- 2-24
FM SORT ---------- 2-24
FM SORT PHASE ---------- 2-16
FM SUPERVISOR ---------- 2-24
FMX ---------- 2-18, 2-24
FORMATTED FILE ---------- 2-1
FORMATTED FILES ---------- 2-22
FR ---------- 2-24
FS INPUT CARD ---------- 3-6
GEOGRAPHIC OPERATOR ---------- 2-12
GEOOP CARD ---------- 3-30
GROUP ---------- 2-23
GROUP CARD ---------- 3-24
GROUPS ---------- 2-2
HOP ---------- 3-36
ICC OPCODE METHOD ---------- 4-35, 4-37
IF ---------- 4-77
INDEX CARD ---------- 3-26
INPUT DESCRIPTOR DECK ---------- 4-41
INPUT FILE ---------- 2-16
INPUT GROUP ---------- 4-21
INPUT GROUP CONTROL FIELD ---------- 4-21
INPUT GROUP CONTROL FIELD LOCATOR CARD ---------- 4-44
INPUT PROCESSING PHASE ---------- 2-16
INPUT PROCESSOR ---------- 2-24

INPUT RECORD ---------- 4-21
INPUT RECORD TYPE ---------- 4-21
INPUT RECORD TYPE CODE ---------- 4-21
INPUT RECORD TYPE CODE LOCATOR CARD ---------- 4-41
INTERNAL FORMAT ---------- 2-16, 4-11
IP ERROR ---------- 4-62
LFM ---------- 4-76
LIST ---------- 4-77, 4-81
LOCATE ---------- 4-77
LOGICAL FILE MAINTENANCE ---------- 4-76
LOOKUP ---------- 4-77, 4-81
MAXL ---------- 4-69
MOVE ---------- 4-78
MULTIPLE SUBSET ENTRIES ---------- 4-27
NADD ---------- 4-70
NCAD ---------- 4-70
NCRF ---------- 4-70
NOCR ---------- 4-70
NOID ---------- 4-70
NOOP ---------- 4-70
NSET ---------- 4-70
NTYP ---------- 4-71
NVST ---------- 4-71
OPCODE POSITION LOCATOR CARD ---------- 4-46
OPERATING SYSTEM ---------- 1-3
OR ---------- 4-78
OUTPUT PROCESSING ---------- 1-4
PARAMETER CHANGES ---------- 4-83
PERIODIC DATA ---------- 2-1, 2-22
PERIODIC FIELD ---------- 2-2, 2-22
PERIODIC FIELDS/GROUPS ---------- 4-26
PERIODIC GROUPS ---------- 2-3
PERIODIC SET ---------- 2-23
PERIODIC SET ONE ---------- 2-6
PERIODIC SET TWO ---------- 2-6
PERIODIC SETS ---------- 2-3
PERIODIC SUBSET ---------- 2-23
PERIODIC SUBSET SEQUENCE NUMBER PSSQN ---------- 4-27
PERIODIC SUBSETS ---------- 2-3
POLYGONS ---------- 2-12
PROGRAMMING SYSTEMS ---------- 1-3
PSER ---------- 4-71
QUANTITATIVE LIMITS ---------- 3-4
RCTL ---------- 4-71
RECORD CONTROL FIELD LOCATOR CARD ---------- 4-49
RECORD ID ---------- 2-4, 2-23
RECORD ID CODE METHOD ---------- 4-35
RETRIEVAL ---------- 1-4

SERVICE PROGRAMS ---------- 1-3
SIGN CONTROL ---------- 3-16
SORT ---------- 4-71
SPECIAL GEOGRAPHIC OPERATOR ---------- 2-25
SPLITTING FIXED FIELDS ---------- 4-25
SPLITTING FIXED GROUPS ---------- 4-26
SSID ---------- 4-71
STATUS ---------- 3-10
STRIP ---------- 4-78
SUB AND TAB CARDS ---------- 3-20
SUBROUTINE VERIFICATION ---------- 3-19
SUBSET ENTRIES ---------- 4-30
SYNONYM TABLE ---------- 2-8
SYNONYMS ---------- 2-8
SYSTEM MONITOR ---------- 1-3
THE CMFLA SAMPLE FILE ---------- 2-6
THE TRANSACTION RECORD LAYOUT ---------- 4-58
TRANSACTION CONFIRMATION LISTING ---------- 4-62
TRANSACTION CONFIRMATION TAPE ---------- 2-17
TRANSACTIONS ---------- 2-13
UPDATE ---------- 4-78
VARIABLE DATA ---------- 2-22
VARIABLE SET ---------- 2-3, 2-6, 2-23, 4-33
VARIABLE SET DATA ---------- 3-5
VSET CARD ---------- 3-25
ZERO SUPPRESSION ---------- 3-13
1410/7010 OPERATING SYSTEMS ---------- 1-3
1410 FFS ---------- 1-2